

BBBBBBBBBBBB		000000000		000000000		TTTTTTTTTTTT		SSSSSSSSSS
BBBBBBBBBBBB		000000000		000000000		TTTTTTTTTTTT		SSSSSSSSSS
BBBBBBBBBBBB		000000000		000000000		TTTTTTTTTTTT		SSSSSSSSSS
BBB	BBB	000	000	000	000	TTT	SSS	
BBB	BBB	000	000	000	000	TTT	SSS	
BBB	BBB	000	000	000	000	TTT	SSS	
BBB	BBB	000	000	000	000	TTT	SSS	
BBB	BBB	000	000	000	000	TTT	SSS	
BBB	BBB	000	000	000	000	TTT	SSS	
BBB	BBB	000	000	000	000	TTT	SSS	
BBBBBBBBBBBB		000	000	000	000	TTT	SSS	SSSSSSSS
BBBBBBBBBBBB		000	000	000	000	TTT	SSS	SSSSSSSS
BBBBBBBBBBBB		000	000	000	000	TTT	SSS	SSSSSSSS
BBB	BBB	000	000	000	000	TTT	SSS	
BBB	BBB	000	000	000	000	TTT	SSS	
BBB	BBB	000	000	000	000	TTT	SSS	
BBB	BBB	000	000	000	000	TTT	SSS	
BBB	BBB	000	000	000	000	TTT	SSS	
BBB	BBB	000	000	000	000	TTT	SSS	
BBBBBBBBBBBB		000000000		000000000		TTT	SSS	SSSSSSSS
BBBBBBBBBBBB		000000000		000000000		TTT	SSS	SSSSSSSS
BBBBBBBBBBBB		000000000		000000000		TTT	SSS	SSSSSSSS

```
CCCCCCCC 000000 NN NN FFFFFFFF IIIIII GGGGGGGG UU UU TTTTTTTTTT LL
CCCCCCCC 000000 NN NN FFFFFFFF IIIIII GGGGGGGG UU UU TTTTTTTTTT LL
CC CC 00 00 NN NN FF IIII GG GG UU UU TT TT LL
CC CC 00 00 NN NN FF IIII GG GG UU UU TT TT LL
CC CC 00 00 NNNN NN FF IIII GG GG UU UU TT TT LL
CC CC 00 00 NNNN NN FF IIII GG GG UU UU TT TT LL
CC CC 00 00 NN NN FFFFFFFF IIII GG GG UU UU TT TT LL
CC CC 00 00 NN NN FFFFFFFF IIII GG GG UU UU TT TT LL
CC CC 00 00 NN NN FFFFFFFF IIII GG GG UU UU TT TT LL
CC CC 00 00 NN NN FFFFFFFF IIII GG GG UU UU TT TT LL
CC CC 00 00 NN NN FFFFFFFF IIII GG GG UU UU TT TT LL
CC CC 00 00 NN NN FFFFFFFF IIII GG GG UU UU TT TT LL
CC CC 00 00 NN NN FFFFFFFF IIII GG GG UU UU TT TT LL
CCCCCCCC 000000 NN NN FF IIIIII GGGGGG UUUUUUUUUU TT LLLLLLLLLL
CCCCCCCC 000000 NN NN FF IIIIII GGGGGG UUUUUUUUUU TT LLLLLLLLLL

LL IIIIII SSSSSSSS
LL IIIIII SSSSSSSS
LL II SSSSSSSS
LL II SSSSSSSS
LL II SSSSSSSS
LL II SSSSSSSS
LL II SSSSSSSS
LL II SSSSSSSS
LL II SSSSSSSS
LL II SSSSSSSS
LL II SSSSSSSS
LL LLLLLLLLLL IIIIII SSSSSSSS
LL LLLLLLLLLL IIIIII SSSSSSSS
```

(1)	434	BOO\$USEACT - Use active parameters
(4)	602	BOO\$CONFIGALL - Auto-configure all adapters
(4)	797	AUTOLOG - AUTO ALL /LOG formatting
(4)	844	SGN\$GET_DEVICE - Locate device database
(4)	944	Reset routines BOO\$RESETLIST and BOO\$CONRESET and BOO\$MSCP_RESET
(4)	1030	BOO\$CONADP - Set connect adapter number
(4)	1154	BOO\$CONNECT - Connect specified device and load driver
(4)	1337	BOO\$LOAD - Load a driver or misc code if not already loaded
(4)	1346	BOO\$RELOAD - Reload a specified driver
(4)	1431	BOO\$GIVEHELP - Print Help information



```

00000001 0000 1      CONFIGSW=1          ; SET SWITCH TO GENERATE CODE USED BY
          0000 2          ; CONFIGURE PROCESS
          0000 3      .IF      NDF,CONFIGSW
          0000 4      .TITLE   SYSGEN - SYSGEN UTILITY AND PARAMETER FILE EDITOR
          0000 5      .IFF
          0000 6      .TITLE   CONFIGUTL - SYSGEN UTILITIES FOR CONFIGURE PROCESS
          0000 7      .ENDC
          0000 8      .IDENT   'V04-002'
          0000 9      *****
          0000 10     *
          0000 11     *  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
          0000 12     *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
          0000 13     *  ALL RIGHTS RESERVED.
          0000 14     *
          0000 15     *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
          0000 16     *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
          0000 17     *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
          0000 18     *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
          0000 19     *  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
          0000 20     *  TRANSFERRED.
          0000 21     *
          0000 22     *  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
          0000 23     *  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
          0000 24     *  CORPORATION.
          0000 25     *
          0000 26     *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
          0000 27     *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
          0000 28     *
          0000 29     *  *****
          0000 30     *
          0000 31     *  ++
          0000 32     *
          0000 33     *  Facility: System generation and initialization
          0000 34     *
          0000 35     *  Abstract: SYSGEN is the main routine to provide all SYSBOOT parameter
          0000 36     *  alteration commands in an online environment.
          0000 37     *
          0000 38     *  Environment:
          0000 39     *
          0000 40     *  Author: RICHARD I. HUSTVEDT, Creation date: 4-MAY-1978
          0000 41     *
          0000 42     *  MODIFIED BY:
          0000 43     *
          0000 44     *  V04-002 WHM0011      Bill Matthews      14-Sep-1984
          0000 45     *  Changed the defaults for the MSCP command.
          0000 46     *
          0000 47     *  V04-001 WHM0010      Bill Matthews      04-Sep-1984
          0000 48     *  Changed IO PRIORITY default for the MSCP command and
          0000 49     *  disallow loading of the MSCP server multiple times.
          0000 50     *
          0000 51     *  V03-023 WHM0009      Bill Matthews      23-Jul-1984
          0000 52     *  Changed defaults for the MSCP command.
          0000 53     *
          0000 54     *  V03-022 WHM0008      Bill Matthews      20-Apr-1984
          0000 55     *  Removed WRITE CURRENT code that wrote the SYSGEN parameters

```

0000	56	:	to SYS.EXE.
0000	57	:	
0000	58	:	V03-021 WHM0007 Bill Matthews 04-Apr-1984
0000	59	:	Added support to write current to write to a seperate
0000	60	:	default system parameter file.
0000	61	:	Added support to use file to accept long ascii sysgen parameters
0000	62	:	
0000	63	:	V03-020 WHM0006 Bill Matthews 14-Mar-1984
0000	64	:	Modify SGN\$GET_DEVICE to take out the I/O database MUTEX and
0000	65	:	raise IPL before calling IOC\$SEARCHALL.
0000	66	:	
0000	67	:	V03-019 WHM0005 Bill Matthews 13-Mar-1984
0000	68	:	Move definition of BOO\$GL_LOAD_ARGS from SYSBOOCMD to
0000	69	:	this module.
0000	70	:	
0000	71	:	V03-018 ACG0399 Andrew C. Goldstein 10-Mar-1984 0:36
0000	72	:	Change check for SS\$ NODEVAVL to SS\$_NOSUCHDEV due to
0000	73	:	rewrite of IOC\$SEARCHDEV.
0000	74	:	
0000	75	:	V03-016 WHM0004 Bill Matthews 23-Feb-1984
0000	76	:	Added support for loading and starting the MSCP server.
0000	77	:	
0000	78	:	V03-015 WHM0003 Bill Matthews 04-Feb-1984
0000	79	:	Added support for ACF\$B_COMBO_VECTOR_OFFSET to clean up support
0000	80	:	of combo style devices.
0000	81	:	
0000	82	:	V03-014 TMK0001 Todd M. Katz 31-Jan-1984
0000	83	:	Change a BSBW to a JSB.
0000	84	:	
0000	85	:	V03-013 WHM0002 Bill Matthews 13-Dec-1983
0000	86	:	Fixed several calls to SGN\$GET_DEVICE to pass the unit number
0000	87	:	to be connected not the maximum units.
0000	88	:	Added support for the new CONNECT command qualifiers
0000	89	:	/CSR_OFFSET and /VECTOR_OFFSET.
0000	90	:	
0000	91	:	V03-012 JLV0312 Jake VanNoy 26-Oct-1983
0000	92	:	Fix bug for microVAX that allows nexus 0 in CONNECT.
0000	93	:	
0000	94	:	V03-011 WHM0001 Bill Matthews 09-Dec-1983
0000	95	:	Changed some bsbw's to jsb's
0000	96	:	
0000	97	:	V03-010 WMC0003 Wayne Cardoza 09-Aug-1983
0000	98	:	Fix loadable code error handling.
0000	99	:	USEACTIVE should be in configutl.
0000	100	:	
0000	101	:	V03-009 WMC0002 Wayne Cardoza 29-Jul-1983
0000	102	:	More features for code loading.
0000	103	:	
0000	104	:	V03-008 WMC0001 Wayne Cardoza 27-Jul-1983
0000	105	:	Support general code loading.
0000	106	:	
0000	107	:	V03-007 MSH0006 Maryann Hinden 24-Jun-1983
0000	108	:	Use \$BOOCMDDEF instead of \$BOODEF.
0000	109	:	
0000	110	:	V03-006 MSH0005 Maryann Hinden 04-May-1983
0000	111	:	Changes to support CONFIGURE process.
0000	112	:	



```

0000 113 : V03-005 MSH0004 Maryann Hinden 13-May-1983
0000 114 : Change some BSBW PUTERROR instructions to JSB instead.
0000 115 :
0000 116 : V03-004 MSH0003 Maryann Hinden 31-Jan-1983
0000 117 : Add support for cluster device names.
0000 118 :
0000 119 : V03-003 TCM0001 Trudy C. Matthews 8-Nov-1982
0000 120 : Use new ADPSL_AVECTOR field in calculation of ACF$W_AVECTOR,
0000 121 : instead of calculating it from the adapter's TR number.
0000 122 :
0000 123 : V03-002 MSH0002 Maryann Hinden 22-Oct-1982
0000 124 : Fix broken BSBW.
0000 125 :
0000 126 : V03-001 MSH0001 Maryann Hinden 30-Sep-1982
0000 127 : Check for DDB$$_UCB = 0.
0000 128 : --
0000 129 :
0000 130 :
0000 131 : Include files:
0000 132 :
0000 133 : $ACFDEF : Define autoconfiguration block
0000 134 : $ADPDEF : Define adapter control block
0000 135 : $BOOCMDDEF : Define SYSGEN command options
0000 136 : $CLIDEF : Define CLI codes and values
0000 137 : $CRBDEF : Define CRB offsets
0000 138 : $DDBDEF : Define DDB offsets
0000 139 : $DYNDEF : Block types
0000 140 : $HLPDEF : Define HELP symbols
0000 141 : $IDBDEF : Define IDB offsets
0000 142 : $IHDEF : Image header offsets
0000 143 : $IPLDEF : Define IPLs
0000 144 : $JPIDEF : $GETJPI definitions
0000 145 : $LBRDEF : Librarian symbols
0000 146 : $OPCDEF : Operator message definitions
0000 147 : $PRDEF : Define processor registers
0000 148 : $PRMDEF : Parameter descriptor definitions
0000 149 : $SBDEF : SCS system block definitions
0000 150 : $SHRDEF : Error codes
0000 151 : $SLVDEF : Loadable code header
0000 152 : $SSDEF : Define system status values
0000 153 : $SYSGMSGDEF : Sysgen messages
0000 154 : $TPADEF : TPARSE definitions
0000 155 : $UCBDEF : Define UCB offsets
0000 156 : $VECDDEF : Define VEC offsets
0000 157 :
0000 158 :
0000 159 : Equated Symbols:
0000 160 :
0000000D 0000 161 : CR=13 : Character value for carriage return
0000000C 0000 162 : FF=12 : Character value for form feed
0000000A 0000 163 : LF=10 : Character value for line feed
00001000 0000 164 : UBA_IOBASE=8*512 : Offset from UBA configuration register
0000 165 : to base of I/O page
0000 166 :
0000 167 : Own Storage
0000 168 :
00000000 0000 169 : .PSECT $$$$000,NOEXE,NOVRT : PSECT to mark lower address

```

Address	Label	Value	Description
00000000	BOO\$LOLIM::	---	Marker definition
00000000	.PSECT	---ZZZ,WRT,PAGE	PSECT to mark upper address limit
00000000	BOO\$HILIM::	---	
00000000	.PSECT	NONPAGED_DATA rd,wrt,noexe,quad	
00000000	BOO\$AB_PATCH::	---	Non-paged Patch area
00000200	.BLKB	512	One page
00002200	BOO\$AB_PRMBUF::	---	Parameter buffer
00002200	.BLKB	512*16	A generous buffer
00002400	BOO\$AB_LOADBUF:	---	Buffer for code loader
00002400	.BLKB	512	
00000000	ACF\$GL_DDB::	0	
00000000	ACF\$GL_UCB::	0	
00000000	ACF\$GL_IDB::	0	
00000000	ACF\$GL_CRB::	0	
00000000	ACF\$GL_LASTDDB::	0	
00000000	ACF\$GL_DPT::	0	
00000000	ACF\$GL_SB::	0	
00000000	BOO\$GL_COMBO VECTOR_OFFSET::	0	Offset to vector from start of combo device's vectors
00000000	BOO\$GL_COMBO CSR_OFFSET::	0	Offset to CSR from start of combo device's CSR
00000000	BOO\$GL_CONADP::	-2	Adapter TR number
FFFFFFFFE	BOO\$GL_CONCREG::	-1	Control register
FFFFFFFFF	BOO\$GL_CONCUNIT::	-1	Controller unit
FFFFFFFFF	BOO\$GL_CONNUMU::	1	Number of Units to configure
00000001	BOO\$GL_CONVECT::	-1	Default value is 1 unit
FFFFFFFFF	BOO\$GL_CONNUMV::	-1	Vector offset
FFFFFFFFF	BOO\$GL_CONAUNIT::	-1	Number of vectors
FFFFFFFFF	BOO\$GL_CONDEV::	-1	Adapter unit
FFFFFFFFF	BOO\$GL_CONDRV::	-1	Device name string address
FFFFFFFFF	BOO\$GL_CONUNITS::	0	Null value
00000000	BOO\$GL_CONSYSID::	0	Driver name string address
00000000	BOO\$GL_CONCRB::	0	Null value
00000000	BOO\$GL_CONFLAGS::	0	Maximum units
00000000	BOO\$GL_NEXTSTR::	0	System ID
00000000	BOO\$GL_CONCRB::	0	quadword
00000000	BOO\$GL_CONCRB::	0	CRB address
00000000	BOO\$GL_CONCRB::	0	Flags
00000000	BOO\$GL_CONCRB::	0	Next string location



00000000	245C	227	BOO\$GL_SELECT::	.LONG 0	: Address of select list
00000000	2460	228	BOO\$GL_SELECT::	.LONG 0	: CLI call back block
	2464	229	BOO\$AL_CLIBLK::	.LONG 0	: Get command call back block
	2464	230	BOO\$AL_CLIBLK::	.LONG 0	: ; Command descriptor address
0000246C	2464	231	BOO\$GQ_CMDESC::	RQTYPE=CLISK_GETCMD	: Prompt string
	2480	232	BOO\$GT_PROMPT::	BOO\$AL_CLIBLK+CLISW_RQSIZE	: Auto-configuration block
00 20 20 3E 4E 45 47 53 59 53 0A 0D	2480	233	BOO\$GT_PROMPT::	.ASCIZ <CR><LF>%SYSGEN> %	: Allocate space for it
	248C	234	BOO\$AL_ACF::	.BLKB ACF\$C_LENGTH	: High and low address limits for lockdown
000024B4	248C	235	BOO\$GQ_LIMITS::	.LONG BOO\$LOLIM	: Lower address bound
	24B4	236	BOO\$GQ_LIMITS::	.LONG BOO\$HILIM-1	: Return address receiver
00000000	24B4	237	BOO\$GQ_RETADR::	.LONG 0,0	: Saved co-routine return address
FFFFFFFF	24B8	238	BOO\$GL_RETSAVE::	.LONG 0	: Facility name descriptor
	24BC	239	FACNAMED::	.LONG FACNAMSZ,FACNAME	: Length of facility name
00000000 00000000	24BC	240	FACNAME::	.ASCIZ /SYSGEN/	: Console block storage
	24C4	241	FACNAMSZ::	.ASCIZ /CSA/	: device name
	24C4	242	CONSNAME::	.ASCIZ /OPA/	: Console terminal device name
00000000	24C8	243	BOO\$GT_OPNAME::	.ASCIZ /OPA/	: Name of RL02 driver
000024D0	24C8	244	BOO\$GT_OPNAME::	.ASCIZ /OPA/	: Name of floppy driver
4E 45 47 53 59 53	24D0	245	BOO\$GT_CVNAME::	.ASCIZ /CVDRIVER/	: Name of TU58 driver
00000006	24D6	246	BOO\$GT_CVNAME::	.ASCIZ /CVDRIVER/	: File spec address
	24D6	247	BOO\$GT_DXNAME::	.ASCIZ /DXDRIVER/	: File spec length
41 53 43 00	24D6	248	BOO\$GT_DXNAME::	.ASCIZ /DXDRIVER/	: Name of TU58 driver
03	24D6	249	BOO\$GT_DDNAME::	.ASCIZ /DDDRIVER/	: File spec address
	24DA	250	BOO\$GT_DDNAME::	.ASCIZ /DDDRIVER/	: File spec length
41 50 4F 00	24DA	251	BOO\$GL_FILEADDR::	.LONG 0	: File spec address
03	24DA	252	BOO\$GL_FILEADDR::	.LONG 0	: File spec length
52 45 56 49 52 44 56 43 00	24DE	253	BOO\$GB_FILELEN::	.BYTE 0	: File spec address
08	24DE	254	BOO\$GB_FILELEN::	.BYTE 0	: File spec length
	24DE	255	BOO\$GL_PARINUSE::	.LONG 0	: File spec address
52 45 56 49 52 44 58 44 00	24E7	256	BOO\$GT_CURRENT::	.ASCIZ /Current/	: File spec length
08	24E7	257	BOO\$GT_CURRENT::	.ASCIZ /Current/	: File spec address
	24E7	258	BOO\$GT_ACTIVE::	.ASCIZ /Active/	: File spec length
52 45 56 49 52 44 44 44 00	24F0	259	BOO\$GT_ACTIVE::	.ASCIZ /Active/	: File spec address
08	24F0	260	BOO\$GT_ACTIVE::	.ASCIZ /Active/	: File spec length
	24F0	261	BOO\$GT_DEFAULT::	.ASCIZ /Default/	: File spec address
	24F0	262	BOO\$GT_DEFAULT::	.ASCIZ /Default/	: File spec length
	24F9	263	BOO\$GT_FILE::	.BLKB 64	: File spec address
00000000	24F9	264	BOO\$GT_FILE::	.BLKB 64	: File spec length
74 6E 65 72 72 75 43 00	24FD	265	HELP_FILE::	.ASCIZ /SYSS\$HELP:SYSGEN.HLB/	: Help library file name
07	24FD	266	HELP_FILE::	.ASCIZ /SYSS\$HELP:SYSGEN.HLB/	: Help library file name
65 76 69 74 63 41 00	24FE	267	HELP_FLAG::	.long hlp\$m_prompt	: Help library file name
06	24FE	268	HELP_FLAG::	.long hlp\$m_prompt	: Help library file name
74 6C 75 61 66 65 44 00	2502	269	HELP_FLAG::	.long hlp\$m_prompt	: Help library file name
07	2502	270	HELP_FLAG::	.long hlp\$m_prompt	: Help library file name
00002559	250A	271	HELP_FLAG::	.long hlp\$m_prompt	: Help library file name
	250A	272	HELP_FLAG::	.long hlp\$m_prompt	: Help library file name
45 48 24 53 59 53 00002561	2511	273	HELP_FLAG::	.long hlp\$m_prompt	: Help library file name
4C 48 2E 4E 45 47 53 59 53	2511	274	HELP_FLAG::	.long hlp\$m_prompt	: Help library file name
3A 50 4C	2511		HELP_FLAG::	.long hlp\$m_prompt	: Help library file name
42	2511		HELP_FLAG::	.long hlp\$m_prompt	: Help library file name
00000001	2511		HELP_FLAG::	.long hlp\$m_prompt	: Help library file name



```

00002580'010E0000' 2578 274 HELP_DESC: .ascid // ; Filled in as pointer
2580 275
2580 276 VALID_PAR_FILE: ; Valid parameter file flag
00000000 2580 277 .LONG 0
2584 278 SAVE_DOT: ; Save dot through USE filespec
00000000 2584 279 .LONG 0
2588 280 FULL_NAME_PTR:: ; Full device name
00000000 2588 281 .LONG 0
258C 282
258C 283 ; MSCP initialization routine default argument list
258C 284
258C 285 MSCP_ARG_LIST:
00000008 258C 286 .LONG 8 ; Number of arguments
00000001 2590 287 .LONG 1 ; Function code(load and start server)
00008000 2594 288 .LONG 32768 ; Default buffer size
00000004 2598 289 .LONG 4 ; Default number of receive credits for each host
0000000F 259C 290 .LONG 15 ; Default number of hosts supported
00000014 25A0 291 .LONG 20 ; Default time out
00000004 25A4 292 .LONG 4 ; Default priority
00001000 25A8 293 .LONG 4096 ; Default for minimum qualifier
00004000 25AC 294 .LONG 16384 ; Default for maximum qualifier
000025BC 25B0 295 .BLKL 3 ; Space for new args
00000030 25BC 296 MSCP_ARG_LIST_SIZE = .-MSCP_ARG_LIST
25BC 297
25BC 298 BOO$GL_LOAD_ARGS:: ; Argument list block loadable code init
000025EC 25BC 299 .BLRB MSCP_ARG_LIST_SIZE ; routine
25EC 300
25EC 301
25EC 302 MSCP_NAME: .ASCIC /MSCP/ ;MSCP server name
25F1 303
25F1 304 ; AUTO ALL /LOG storage
25F1 305
25F1 306 CTRSTR_AUTOLOG: .ascid / !AC!UW/
25FF
55 21 43 41 21 20 000025F9'010E0000' 2600 307 CTRSTR_AUTOLOG_UNIT: .ascid /,!UW/
57 55 21 2C 00002608'010E0000' 260C 308 Outlen_unit: .long 0
00000000 2610 309 Outlen: .long 0
00002628 2614 310 Boo$gt_save_devname: .blkb 20
00002630'010E0000' 2628 311 outbuf: .ascid //
00002694 2630 312 outbuf_str: .blkb 100
2694 313
2694 314 ; Send operator message data
2694 315
2694 316 OPERGETJPI: ; $GETJPI item list
0004 2694 317 .WORD 4 ; Buffer length
0319 2696 318 .WORD JPIS_PID ; Process ID code
000026B0' 2698 319 .ADDRESS OPERMSGPID ; Buffer address
00000000 269C 320 .LONG 0 ; Don't return length
00000000 26A0 321 .LONG 0 ; List terminator
26A4 322
26A4 323 OPERMSGVEC: ; $PUTMSG message vector
0003 26A4 324 .WORD 3 ; Argument count
000F 26A6 325 .WORD ^B1111 ; Default message flags
26A8 326 OPERMSGID:
00000000 26A8 327 .LONG 0 ; Message ID
26AC 328 OPERMSGFAO:

```

```

0001 26AC 329 .WORD 1 ; FA0 argument count
0000 26AE 330 .WORD 0 ; No new message flags
26B0 331 OPERMSGPID: ; PID of this process
00000000 26B0 332 .LONG 0 ; File specification
26B4 333 OPERMSGNAM:
000026B8' 26B4 334 .ADDRESS OPERNAMDESC
26B8 335
26B8 336 OPERNAMDESC:
00000000 00000000 26B8 337 .LONG 0,0
26C0 338
26C0 339 OPERMSG: ; Message descriptor
00000000 26C0 340 .LONG 0
000026C8' 26C4 341 .ADDRESS 0 OPERMSGBUF
26C8 342
26C8 343 OPERMSGBUF: ; Message buffer
00000103 26C8 344 .LONG OPC$_RQ_RQST!<OPCSM_NM_CENTRL@8> ; Message type and target
00000000 26CC 345 .LONG 0 ; No reply message
26D0 346 OPERMSGTXT: ; Message text
000027D0 26D0 347 .BLKB 256
27D0 348
27D0 349 .IF NDF,CONFIGSW ; SYSGEN-specific code
27D0 350 .PAGE
27D0 351 .SBTTL BOO$USEFILE - Use parameter file
27D0 352
27D0 353 ;++
27D0 354 ; Functional description:
27D0 355 BOO$USEFILE reads the specified file in response to the USE
27D0 356 command and merges all of the values specified in that file into
27D0 357 the working copy of the parameter values. This is accomplished
27D0 358 by looking up each value specified and merging the associated
27D0 359 value.
27D0 360 ; Calling sequence:
27D0 361 CALLG arglist,BOO$USEFILE
27D0 362
27D0 363 ; Input Parameters:
27D0 364 TPA$L_TOKENCNT(AP) - Length of file name string
27D0 365 TPA$L_TOKENPTR(AP) - Address fo file name string
27D0 366 ; Output Parameters:
27D0 367 R0 - Completion status code
27D0 368
27D0 369 ;--
27D0 370
27D0 371 .PSECT PAGED_CODE rd,nowrt,exe,long
27D0 372
27D0 373 .Entry BOO$USEFILE, ^M<R2,R3,R4,R5,R6,R7,R8,R9> ; Entry mask
27D0 374
27D0 375
27D0 376 BBSS #EXESV WRITESYSPARAMS,- ; Use a file => write current needed
27D0 377 G^EXESGL_DYNAMIC_FLAGS,1$;
27D0 378 1$:
27D0 379 MOVL BOO$GL_DOT,L^SAVE_DOT ; Save dot
27D0 380 MOVAB TPA$L_TOKENCNT(AP),R7 ; Set address of file name descriptor
27D0 381 BSBW BOO$FIOPEN ; Open specified file
27D0 382 BLBS R0,20$ ; Continue if success
27D0 383 10$: MOVZWL #1,R0 ; Force success
27D0 384 RET
27D0 385 20$: MOVAB BOO$AB_PRMBUF,R6 ; Set address of parameter buffer

```



```

27D0 386      MOVL      #16,R9          : Set size of buffer
27D0 387      BSBW      BOO$READFILE    : Read file content into parameter buffer
27D0 388      BLBC      R0,10$          : Exit if error
27D0 389      MOVAB     BOO$AB PRMBUF,R8 : Init pointer to parameter buffer
27D0 390      MOVCS     #32,(R8),EX$GT_STARTUP : Set startup command file name
27D0 391      ADDL      #32,R8          : and advance buffer pointer
27D0 392      CLRL      VALID_PAR_FILE   : Initialize valid parameter file flag
27D0 393 30$:    TSTL      (R8)          : Check for end of list
27D0 394      BEQL      DONE            : Branch if yes
27D0 395      MOVZBL    (R8),TPA$L_TOKENCNT(AP) : Set token count for search
27D0 396      MOVAB     1(R8),TPA$L_TOKENPTR(AP) : And address of string
27D0 397      ADDL      #16,R8          : Advance to value
27D0 398      MOVL      (R8)+,TPA$L_NUMBER(AP) : Set number
27D0 399      CALLG     (AP),L^BOO$SEARCH : Search for parameter
27D0 400      BLBC      R0,30$          : Next parameter if not found
27D0 401      MOVL      #1,VALID_PAR_FILE : Indicate valid parameter file
27D0 402      MOVL      TPA$L_PARAM(AP),R4 : Get a pointer to the parameter descriptor
27D0 403      BBC       #PRM$V_ASCII,PRM$L_FLAGS(R4),40$ : Branch if not an ascii parameter
27D0 404      MOVAL     -(R8),TPA$L_TOKENPTR(AP) : Get a pointer to the parameter value
27D0 405      MOVZBL    PRM$L_SIZE(R4),R0 : Get parameter size in bits
27D0 406      ASHL      #-3,R0,R0       : Set parameter size
27D0 407      MOVZBL    R0,TPA$L_TOKENCNT(AP) :
27D0 408      ADDL2     #3,R0           : Round size up to the next longword
27D0 409      BICL2     #3,R0           :
27D0 410      ADDL2     R0,R8           : Advance past value
27D0 411      CALLG     (AP),W^BOO$SETASCII : Set the value of the parameter
27D0 412      BRW       30$            : Continue with the next parameter
27D0 413 40$:    CALLG     (AP),L^BOO$SETVALUE : Set value of parameter
27D0 414      BRW       30$            : Continue with next parameter
27D0 415 DONE:    BSBW      BOO$FILCLOSE : Close the file
27D0 416      BLBS      VALID_PAR_FILE,10$ : If LBS, valid parameter file
27D0 417      MOVL      #SYSG$NOTPARAM,R0 : Set error
27D0 418      BRB       20$            : Branch
27D0 419 10$:    :
27D0 420      :
27D0 421      : Set file name in BOO$GL_PARINUSE
27D0 422      :
27D0 423      MOVAL     BOO$GT_FILE,R8    : Set address of String
27D0 424      MOVL      R8,BOO$GL_PARINUSE : Set address
27D0 425      MOVZBL    BOO$GB_FILELEN,(R8) : Set count
27D0 426      MOVCS     (R8),@BOO$GL_FILEADDR,- :
27D0 427      1(R8)      : Move string
27D0 428      :
27D0 429      MOVZWL    #SS$ NORMAL,R0    : Return success
27D0 430 20$:    MOVL      L^SAVE_DOT,BOO$GL_DOT : Restore dot
27D0 431      RET      :
27D0 432      .ENDC     : End of SYSGEN-specific code

```

```

27D0 434 .SBTTL BOO$USEACT - Use active parameters
27D0 435 :++
27D0 436 : Functional description:
27D0 437 : This routine copies the parameter values from the running
27D0 438 : system to the working copy of the parameter values.
27D0 439 : Calling sequence:
27D0 440 :
27D0 441 : CALLS #0,BOO$USEACT
27D0 442 :
27D0 443 : Input parameters:
27D0 444 : None
27D0 445 : Output Parameters:
27D0 446 : R0 - Completion status code
27D0 447 :--
27D0 448
003C 27D0 449 .Entry BOO$USEACT,*M<R2,R3,R4,R5>
27D2 450
27D2 451 MOV C3 #EXESC_SYSPARSZ,- : Move parameters
27D6 452 MMG$A_SYSPARAM,EXE$A_SYSPARAM
27E0 453 MOVAL BOO$GT_ACTIVE,-
27E4 454 BOO$GL_PARINUSE : Set parameter in use
27E7 455 MOVL #1,R0 : Return success
27EA 456 RET
27EB 457 .IF NDF,CONFIGSW : SYSGEN-specific code

```



```

27EB 459 .SBTTL BOO$WRTACT - Write parameters to system
27EB 460 :++
27EB 461 : Functional Description:
27EB 462 : This routine writes the parameters in the working parameter
27EB 463 : buffer to the system's parameter area. Only dynamic
27EB 464 : parameters are copied.
27EB 465 :
27EB 466 : Calling Sequence:
27EB 467 : CALLS #0,BOO$WRTACT
27EB 468 :
27EB 469 : Input Parameters:
27EB 470 : None
27EB 471 :
27EB 472 : Output Parameters:
27EB 473 : R0 - Completion status code
27EB 474 :--
27EB 475
27EB 476 .PSECT NONPAGED_CODE rd,nowrt,exe,long
27EB 477
27EB 478 .Entry BOO$WRTACT, ^M<>
27EB 479
27EB 480 $CMKRNLS B^10$(AP) : Do it in kernel mode
27EB 481 BLBC RO,1$ : If LBC, error
27EB 482 JSB BOO$SENDOPER : Notify operator of WRITE ACTIVE
27EB 483 .LONG SYSG$_WRITEACT
27EB 484 BLBS RO,5$ : If LBS, success
27EB 485 1$: JSB PUTERROR : Report error
27EB 486 MOVL #1,R0 : Force success
27EB 487 5$: RET
27EB 488
27EB 489 10$: .WORD ^M<R2,R3,R4,R5>
27EB 490 MOVAB L^BOO$A PRMBLK,R5 : Get base of parameter blocks
27EB 491 DSBINT #IPL$_SCHED : Raise IPL to prevent being unscheduled
27EB 492 : (Assumes pages are locked in W.S.)
27EB 493
27EB 494 ASSUME PRM$L_ADDR EQ 0
27EB 495
27EB 496 20$: MOVL PRM$L_ADDR(R5),R3 : Get address of parameter
27EB 497 BEQL 40$ : Reached the end
27EB 498 BBC #PRM$V_DYNAMIC,- : Branch if this is not a
27EB 499 PRM$L_FLAGS(R5),30$ : dynamic parameter
27EB 500 MOVZBL PRM$B_POS(R5),R1 : Get position of parameter
27EB 501 EXTZV R1,PRM$B_SIZE(R5),(R3),R2 : Extract parameter value
27EB 502 MOVAB L^EXESA_SYSPARAM,R0 : Get address of working buffer
27EB 503 SUBL R0,R3 : Get parameter offset
27EB 504 INSV R2,R1,PRM$B_SIZE(R5),- : Store in system
27EB 505 L^MMGSA_SYSPARAM(R3)
27EB 506
27EB 507 30$: ADDL #PRM$C_LENGTH,R5 : Point to next parameter block
27EB 508 BRB 20$ : Repeat
27EB 509
27EB 510 : Copy dynamic flags from default flags to R0
27EB 511
27EB 512 40$: BICL3 #^C<PRM$M_DYNFLAGS>,- :
27EB 513 MMGSA_SYSPARAM+<EXE$GL_DEFFLAGS-EXESA_SYSPARAM>,R0 :
27EB 514 BICL #PRM$M_DYNFLAGS,- : Clear dynamic flags in real flags
27EB 515 EXE$GL_FLAGS

```

CONFIGUTL  
V04-002

- SYSGEN UTILITIES FOR CONFIGURE PROCESS H 2  
BOO\$USEACT - Use active parameters

15-SEP-1984 23:46:56 VAX/VMS Macro V04-00  
14-SEP-1984 16:09:11 [BOOTS.SRC]SYSGEN.MAR;3

Page 11  
(2)

27EB	516	BISL	RO,EXE\$GL_FLAGS	: Set dynamic flags in real flags
27EB	517			
27EB	518	ENBINT		: Lower IPL
27EB	519	MOVL	#1,RO	: Set success
27EB	520	RET		



```

27EB 522      .SBTTL BOO$WRTCUR - Write Current Parameters
27EB 523      :++
27EB 524      : Functional Description:
27EB 525      : This routine writes the parameters from the working parameter
27EB 526      : buffer to the system parameter file on disk. They will take effect the
27EB 527      : next time the system is booted.
27EB 528      :
27EB 529      : Calling Sequence:
27EB 530      : CALLS #0,BOO$WRTCUR
27EB 531      :
27EB 532      : Input parameters:
27EB 533      : None
27EB 534      :
27EB 535      : Output Parameters:
27EB 536      : R0 - Completion status code
27EB 537      :--
27EB 538
27EB 539      .PSECT PAGED_CODE      rd,nowrt,exe,long
27EB 540
27EB 541      .Entry BOO$WRTCUR, *M<R2,R3,R4,R5,R6,R7,R8,R9>
27EB 542
27EB 543      BBCC      #EXESV WRITESYSPARAMS,- : Don't do WRITE CURRENT again in startup
27EB 544      G*EXESGL DYNAMIC FLAGS,10$:
27EB 545      10$:      MOVAB      BOO$GT SYSPARNAME,R0 : Get address of system .PAR file name
27EB 546      MOVZBL      (R0)+,TPASL TOKENCNT(AP) : Set up for call to BOO$WRTSYSPARFILE
27EB 547      MOVL      R0,TPASL TOKENPTR(AP) :
27EB 548      CALLG      (AP),G*BOO$WRTSYSPARFILE : Call the routine to write out the file
27EB 549      BLBC      R0,20$ : Branch if error
27EB 550      BSBW      BOO$SENDOPER : Notify operator of WRITE CURRENT
27EB 551      .LONG      SYSG$_WRITECUR
27EB 552      BLBS      R0,30$ : If LBS, success
27EB 553      20$:      BSBW      PUTERROR : Report error
27EB 554      30$:      MOVL      #1,R0 : Return success
27EB 555      RET
27EB 556

```

```

27EB 558      .SBTTL BOO$SENDOPR - Output facility error message to operator
27EB 559      :
27EB 560      : Functional Description:
27EB 561      : BOO$SENDOPR outputs an error message to the operator.
27EB 562      :
27EB 563      : Calling Sequence:
27EB 564      : BSBW BOO$SENDOPR
27EB 565      : .LONG <msg-id>
27EB 566      :
27EB 567 BOO$SENDOPR::
27EB 568      MOVL @ (SP), OPERMSGID ; Put message ID in vector
27EB 569      ADDL2 #4, (SP) ; Advance return address
27EB 570      $GETJPI S ITMLST=OPERGETJPI ; Get process ID
27EB 571      BLBC R0, 10$ ; If LBC, error
27EB 572      MOVL #3, OPERMSGVEC ; Assume WRITE ACTIVE
27EB 573      MOVL #1, OPERMSGFAO
27EB 574      CLRL OPERMSGNAM
27EB 575      CMPL #SYSG$_WRITECUR, OPERMSGID ; WRITE CURRENT ?
27EB 576      BNEQ 5$ ; If NEQ, no
27EB 577      INCL OPERMSGVEC ; Set up WRITE CURRENT
27EB 578      INCL OPERMSGFAO
27EB 579      MOVAB OPERNAMDESC, OPERMSGNAM
27EB 580      MOVZBL RIO_INPNAM+NAM$B_RSL, OPERNAMDESC; Build descriptor
27EB 581      MOVL RIO_INPNAM+NAM$B_RSA, OPERNAMDESC+4
27EB 582 5$: SPUTMSG S - ; Get and format message
27EB 583      MSGVEC=OPERMSGVEC, -
27EB 584      ACTRTN=666$
27EB 585      BLBC R0, 10$ ; If LBC, error
27EB 586      $SNDOPR S MSGBUF=OPERMSG
27EB 587      BLBS R0, 20$ ; If LBS, success
27EB 588 10$: BSBW PUTERROR ; Report error
27EB 589      MOVL #1, R0 ; Force success
27EB 590 20$:
27EB 591      RSB
27EB 592 666$:
27EB 593      .WORD *M<R2,R3,R4,R5>
27EB 594      MOVQ @4(AP), R0 ; Get string descriptor
27EB 595      ADDL3 #OPCSL MS TEXT, R0, OPERMSG; Store total operator message size
27EB 596      MOVC3 R0, (R1), OPERMSGTXT ; Copy text to operator message buffer
27EB 597      CLRL R0 ; Prevent message output to SYS$OUTPUT
27EB 598      RET
27EB 599
27EB 600      .ENDC ; End of SYSGEN-specific code

```

```

27EB 602 .SBTTL BOO$CONFIGALL - Auto-configure all adapters
27EB 603 :++
27EB 604 : Functional Description:
27EB 605 : BOO$CONFIGALL is called to implement the "AUTOCONFIGURE ALL"
27EB 606 : command. All standard devices supported by VAX/VMS will be
27EB 607 : located and connected for use with any necessary drivers being
27EB 608 : loaded.
27EB 609 :
27EB 610 : Calling Sequence:
27EB 611 : CALLG ARGLIST,BOO$CONFIGALL
27EB 612 :
27EB 613 : Output parameters:
27EB 614 : R0 - Completion status code
27EB 615 : --
27EB 616 :
00000000 617 .PSECT NONPAGED_CODE rd,nowrt,exe,long
0000 618
OFFC 0000 619 .Entry BOO$CONFIGALL, ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ; Entry mask
0002 620
08 00000000'EF 00000000'8F E1 0002 621 BBC #EXESV_NOAUTOCNF,EXESGL_DEFFLAGS,5$; do we allow auto configure
50 007C8002 8F D0 000E 622 MOVL #SYSGS_NOAUTOCNF,R0 ; Give them a no autoconfigure error
04 0015 623 RET ; and return
0016 624
FFE7' 30 0016 625 5$: BSBW BOO$LOCK_GEN ; Lock SYSGEN database
07 50 E8 0019 626 BLBS R0,7$ ; If no error, continue
00000000'EF 16 001C 627 JSB PUTERROR
04 0022 628 RET
0023 629
FFDA' 30 0023 630 7$: BSBW IOC$AUTORESET ; Reset controller characters for device
0026 631 names
5B D4 0026 632 CLRL R11 ; Indicate no ADP address yet
5B DD 0028 633 10$: PUSHL R11 ; Set as argument
000000B2'EF 01 FB 002A 634 CALLS #1,NEXTADP ; Get next ADP address
29 50 E9 0031 635 BLBC R0,CONFIG_EXIT ; Branch if error (NOPRIV)
5B 51 D0 0034 636 MOVL R1,R11 ; Check return status
10 18 0037 637 BGEQ 20$ ; Branch if done
5B DD 0039 638 PUSHL R11 ; Set as ADP argument
0103'CF 01 FB 003B 639 CALLS #1,W^CONFIGADP ; Configure the entire adapter
E5 50 E8 0040 640 BLBS R0,10$ ; Continue if no error
00000000'EF 16 0043 641 JSB PUTERROR ; Report error
50 01 D0 0049 642 20$: MOVL #1,R0 ; Set success
004C 643
09 00000000'EF 0C E1 004C 644 BBC #BOOCMD$V AUTOLOG,L^BOO$GL CMDOPT,CONFIG_EXIT ; Branch if not /LOG
00002614'EF D4 0054 645 CLRL BOO$GT_SAVE_DEVNAME ; Clear name
01A9 30 005A 646 BSBW AUTOLOG ; Output last line if there is one
005D 647
005D 648 CONFIG_EXIT:
50 DD 005D 649 PUSHL R0 ; Save status
FF9E' 30 005F 650 BSBW BOO$UNLOCK_GEN ; Unlock SYSGEN database
06 50 E8 0062 651 BLBS R0,35$ ; If no error, continue
00000000'EF 16 0065 652 JSB PUTERROR ; Give error message
50 8ED0 006B 653 35$: POPL R0 ; Restore status
04 006E 654 RET
006F 655
OFFC 006F 656 .Entry BOO$CONFIGONE, ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ; Entry mask
0071 657
FF8C' 30 0071 658 BSBW BOO$LOCK_GEN ; Lock SYSGEN database

```



```

07 50 E8 0074 659 BLBS R0,5$ ; If no error, continue
00000000'EF 16 0077 660 JSB PUTERROR ; Give error message
04 007D 661 RET ;
FF7F' 30 007E 662 5$: BSBW IOCS$AUTORESET ; Reset controller characters for device
0081 664 ; device names
1C AC DD 0081 665 PUSHL TPASL_NUMBER(AP) ; Set TR number of adapter
DA'AF 01 FB 0084 666 CALLS #1,B*LOCADP ; Locate adapter control block
D2 50 E9 0088 667 BLBC R0,CONFIG_EXIT ; Branch if error (NOPRIV)
51 DD 008B 668 PUSHL R1 ; Set as argument to CONFIGADP
0D 13 008D 669 BEQL 10$ ; Done if no adapter
03'AF 01 FB 008F 670 CALLS #1,B*CONFIGADP ; Configure adapter
06 50 E8 0093 671 BLBS R0,10$ ; Continue if no error
00000000'EF 16 0096 672 JSB PUTERROR ; Give error status
50 01 D0 009C 673 10$: MOVL #1,R0 ; Set success for parse
B6 00000000'EF 0C E1 009F 674 BBC #BOOCMD$V_AUTOLOG,L*BOOS$GL_CMDOPT,CONFIG_EXIT ; Branch if not /LOG
00002614'EF D4 00A7 675 CLRL BOOS$GT_SAVE_DEVNAME ; Clear name
0156 30 00AD 676 BSBW AUTOLOG ; Output last line if there is one
AB 11 00B0 677 20$: BRB CONFIG_EXIT ;
00B2 678 ;
00B2 679 NEXTADP: ; Return next ADP address in R0
0000 00B2 680 .WORD 0 ; Null entry mask
04 00B4 681 $CMEXEC,S B*10$(AP) ; Call real routine in exec mode
00C0 682 RET ;
00C1 683 ;
0000 00C1 684 10$: .WORD 0 ; Null entry mask
51 04 AC D0 00C3 685 MOVL 4(AP),R1 ; Get current address
06 13 00C7 686 BEQL 20$ ; 0 => start of list
51 04 A1 D0 00C9 687 MOVL ADPSL_LINK(R1),R1 ; Flink onward
07 11 00CD 688 BRB 30$ ;
51 00000000'EF D0 00CF 689 20$: MOVL IOCS$GL_ADPLIST,R1 ; Return head of list
50 01 D0 00D6 690 30$: MOVL #1,R0 ;
04 00D9 691 RET ;
00DA 692 ;
00DA 693 LOCADP: ; Return address of ADP for TR number
0000 00DA 694 .WORD 0 ;
00DC 695 $CMEXEC,S B*5$(AP) ; Call routine in exec mode
04 00E8 696 RET ;
00E9 697 ;
0000 00E9 698 5$: .WORD 0 ; Null entry mask
51 FFFFFFFC'EF 9E 00EB 699 MOVAB IOCS$GL_ADPLIST-ADPSL_LINK,R1 ; Set starting address
51 04 A1 D0 00F2 700 10$: MOVL ADPSL_LINK(R1),R1 ; Flink onward
07 13 00F6 701 BEQL 20$ ; Done if at end
0C A1 04 AC B1 00F8 702 CMPW 4(AP),ADPSW_TR(R1) ; Is this the specified TR?
F3 12 00FD 703 BNEQ 10$ ; No, try another
50 01 D0 00FF 704 20$: MOVL #1,R0 ;
04 0102 705 RET ;
0103 706 ;
00FC 0103 707 .Entry CONFIGADP, ^M<R2,R3,R4,R5,R6,R7>; Entry mask
0105 708 ;
000024C4'EF D4 0105 709 CLRL BOOS$GL_RETSAVE ; Zap return address for initial call
10 00000000'EF 06 E1 010B 710 BBC #BOOCMD$V_SELECT,L*BOOS$GL_CMDOPT,10$ ; Mutually exclusive - test
08 00000000'EF 07 E1 0113 711 BBC #BOOCMD$V_EXCLUDE,L*BOOS$GL_CMDOPT,10$ ; to make sure one bit clear
50 007C808A 8F D0 011B 712 MOVL #SYSG$_CONFQUAL,R0 ; Conflicting qualifiers
04 0122 713 RET ;
0123 714 ;
0171'CF 6C FA 0123 715 10$: CALLG (AP),W*50$ ; Call configure one device

```

```

09 50 E8 0128 716 BLBS R0,20$ ; Branch if not done with this adapter
50 24 B1 012B 717 CMPW #55$_NOPRIV,R0 ; Was there a privilege error
03 13 012E 718 BEQL 15$ ; Yes, branch
50 01 D0 0130 719 MOVL #1,R0 ; Set success
04 0133 720 15$: RET ; and return
0134 721
55 0000248C'EF 9E 0134 722 20$: MOVAB BOOS$_ACF,R5 ; Set address of arguments describing device
0138 723
1C A5 B4 013B 724 CLRW ACF$_MAXUNITS(R5) ; Always use driver specified max units
56 00002460'EF D0 013E 725 MOVL L^BOOS$_SELECT,R6 ; Get pointer to select list
06 13 0145 726 BEQL 35$ ; Branch if null
0087 30 0147 727 BSBW SELECT ; Check select/exclude string
D6 50 E9 014A 728 BLBC R0,10$ ; Branch if device is not to be configured
014D 729
11 0B A5 03 E0 014D 730 35$: BBS #ACF$_NOLOAD_DB,ACF$_AFLAG(R5),38$ ; Branch if not loading database
09 00000000'EF 0C E1 0152 731 BBC #BOOS$_AUTOLOG,L^BOOS$_CMDOPT,38$ ; Branch if not logging
00A9 30 015A 732 BSBW AUTOLOG ; Branch to output log
03 50 E8 015D 733 BLBS R0,38$ ; Branch if no error
FE9D' 30 0160 734 BSBW PUTERROR ; Give error message
0163 735
0000'CF 65 FA 0163 736 38$: CALLG (R5),W^IOGEN$LOADER ; Load database and driver if necessary
B8 50 E8 0168 737 BLBS R0,10$ ; Branch if no error
FE92' 30 016B 738 BSBW PUTERROR ; Give error message
FFB2 31 016E 739 BRW 10$ ; continue loop
0171 740
0000 0171 741 50$: .WORD 0 ;
0173 742 $CMKRNL,S B^55$,(AP) ; Call auto configure in kernel mode
04 017F 743 RET ;
0180 744
OFFC 0180 745 55$: .WORD ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;
50 000024C4'EF D0 0182 746 MOVL BOOS$_RETSAVE,R0 ; Get saved return address
07 12 0189 747 BNEQ 60$ ; Branch if one present
50 00000000'EF 9E 018B 748 MOVAB IOC$AUTOCONFIG,R0 ; Else use main entry point
50 DD 0192 749 60$: PUSHL R0 ; Stack call back address
58 04 AC D0 0194 750 MOVL 4(AP),R8 ; Get address of ADP
56 68 D0 0198 751 MOVL ADP$_CSR(R8),R6 ; Get Configuration register address
57 0000248C'EF 9E 019B 752 MOVAB BOOS$_ACF,R7 ; Address of configuration control block
01A2 753 SETIPL #31 ; Disable interrupts
9E 16 01A5 754 JSB @($P)+ ; Call Auto configuration code
01A7 755 SETIPL #0 ; Enable interrupts
000024C4'EF 8E D0 01AA 756 MOVL ($P)+,BOOS$_RETSAVE ; Save return
06 50 E8 01B1 757 BLBS R0,70$ ; Continue if another device
000024C4'EF D4 01B4 758 CLRL BOOS$_RETSAVE ; Else clear return
01BA 759
0D 0B A7 03 E1 01BA 760 70$: BBC #ACF$_NOLOAD_DB,ACF$_AFLAG(R7),80$ ; Branch if loading database
7E 12 A7 3C 01BF 761 MOVZWL ACF$_UNIT(R7),-($P) ; Get unit number
14 A7 DD 01C3 762 PUSHL ACF$_DEVNAME(R7) ; Get device name
010B 30 01C6 763 BSBW SGN$GET_DEVICE_LOCK_IODB ; Get device database
5E 08 C0 01C9 764 ADDL2 #8,$P ; Clear stack
04 01CC 765 80$: RET ; And return
50 01 3C 01CD 766 90$: MOVZWL #1,R0 ; Set success status
04 01D0 767 RET ; and return
01D1 768
01D1 769 ; SELECT - decide whether current device name is one of those either
01D1 770 ; specified in /SELECT or /EXCLUDE
01D1 771
01D1 772 ; Returns: R0 = 1 ==> configure device

```

```

                                01D1 773 :      R0 = 0  ==> don't configure device
                                01D1 774 :
                                01D1 775 :
                                01D1 776 SELECT:
57 14 A5 D0 01D1 777 10$:  MOVL  ACF$$_DEVNAME(R5),R7      ; Get pointer to device name
                                01D5 778      MOVZBL (R6)+,R4      ; Get length of select entry
                                01D8 779      BEQL  20$      ; End of list, no match
                                01DA 780      CMPB  (R7)+,R4      ; Compare with device entry
                                01DD 781      BLSS  15$      ; Branch if select longer than device
67 66 54 29 01DF 782      CMPC3 R4,(R6),(R7)      ; Do we have a match?
                                01E3 783      BEQL  40$      ; Yes, check SELECT or EXCLUDE
                                01E5 784 15$:  ADDL  R4,R6      ; Advance to next entry in select list
                                01E8 785      BRB   10$      ; And try again
                                01EA 786
                                01EA 787 20$:  CLRL  R0      ; Assume don't configure
03 00000000'EF 50 D4 01EA 787 20$:  CLRL  R0      ; Assume don't configure
                                01EC 788      BBC   #BOOS$V_EXCLUDE,BOOS$GL_CMDOPT,30$ ; Branch if SELECT
                                01F4 789      MOVL  #1,R0      ; EXCLUDE - configure device
                                01F7 790 30$:  RSB
                                01F8 791
                                01F8 792 40$:  CLRL  R0      ; Assume don't configure
03 00000000'EF 50 D4 01F8 792 40$:  CLRL  R0      ; Assume don't configure
                                01FA 793      BBS   #BOOS$V_EXCLUDE,BOOS$GL_CMDOPT,50$ ; Branch if EXCLUDE
                                0202 794      MOVL  #1,R0      ; SELECT - configure device
                                0205 795 50$:  RSB

```



```

0206 797 .SBTTL AUTOLOG - AUTO ALL /LOG formatting
0206 798
0206 799 AUTOLOG::
55 0000248C'EF 9E 0206 800 MOVAB BOO$AL_ACF,R5 ; Address of configuration control block
56 14 A5 D0 020D 801 MOVL ACF$L_DEVNAME(R5),R6 ; Get address of current device
57 86 9A 0211 802 MOVZBL (R6)+,R7 ; Get count and addr.
00002614'EF 66 57 29 0214 803 CMPC3 R7,(R6),BOO$GT_SAVE_DEVNAME ; Compare to previous string
66 57 29 021C 804 BNEQ 50$ ; Branch if new device
39 12 021E 805
021E 806 $FAO_S CTRSTR=CTRSTR_AUTOLOG_UNIT,- ; Format Unit Number
021E 807 OUTBUF=OUTBUF,-
021E 808 OUTLEN=OUTLEN_UNIT,-
021E 809 P1=ACF$W_CUNIT(R5)
03 50 E8 023A 810 BLBS R0,40$ ; Branch if OK
0081 31 023D 811 BRW 100$ ; Branch if error
2610'CF 260C'CF C0 0240 812
262C'CF 260C'CF C0 0247 813 40$: ADDL2 W^OUTLEN_UNIT,W^OUTLEN ; Add to total length
2628'CF 260C'CF A2 024E 814 ADDL2 W^OUTLEN_UNIT,W^OUTBUF+4 ; Add to descriptor
6A 11 0255 815 SUBW2 W^OUTLEN_UNIT,W^OUTBUF ; Subtract from length
0257 816 BRB 100$ ; Return with success
2610'CF D5 0257 817
21 13 025B 818 50$: TSTL W^OUTLEN ; Is this a first call to this routine?
025D 819 BEQL 70$ ; Branch if yes
262C'CF 2630'CF DE 025D 820
0000'CF 2610'CF B0 0264 821 MOVAL W^OUTBUF_STR,W^OUTBUF+4 ; reset descriptor
00C0'CF 28 0266 822 MOVW W^OUTLEN,W^RIO$GW_OUTLEN ; Length of string
2630'CF 026B 823 MOVC3 W^RIO$GW_OUTLEN,-
0000'CF 026F 824 W^OUTBUF_STR,-
0272 825 W^RIO$AB_BUFFER ; Move text into global buffer
0275 826
00000000'EF 16 0275 827 JSB RIO$OUTPUT_LINE
43 50 E9 027B 828 BLBC R0,100$ ; Branch on error
027E 829
2628'CF 0064 8F B0 027E 830 70$: MOVW #100,W^OUTBUF ; Set full buffer length
00002614'EF 66 57 28 0285 831 MOVC3 R7,(R6),BOO$GT_SAVE_DEVNAME ; Save new devname
55 0G00248C'EF 9E 028D 832 MOVAB BOO$AL_ACF,R5 ; Reset R5
0294 833 $FAO_S CTRSTR=CTRSTR_AUTOLOG,- ; Format device name
0294 834 OUTBUF=OUTBUF,-
0294 835 OUTLEN=OUTLEN,-
0294 836 P1=ACF$L_DEVNAME(R5),-
0294 837 P2=ACF$W_CUNIT(R5)
262C'CF 2610'CF C0 02B3 838 ADDL2 W^OUTLEN,W^OUTBUF+4 ; Add to descriptor
2628'CF 2610'CF A2 02BA 839 SUBW2 W^OUTLEN,W^OUTBUF ; Subtract from length
02C1 840 ; Return with FAO status
05 02C1 841 100$: RSB
02C2 842

```

```

02C2 844 .SBTTL SGN$GET_DEVICE - Locate device database
02C2 845
02C2 846
02C2 847 Inputs:
02C2 848 4(SP) - Address of Device name in ascic format
02C2 849 8(SP) - Unit number
02C2 850
02C2 851 Outputs:
02C2 852 (Any of these are 0 if the data block doesn't exist)
02C2 853 ACF$GL_DDB - Address of DDB
02C2 854 ACF$GL_UCB - Address of UCB
02C2 855 ACF$GL_IDB - Address of IDB
02C2 856 ACF$GL_CRB - Address of CRB
02C2 857 ACF$GL_SB - Address of SB
02C2 858 ACF$GL_LASTDDB - If ACF$GL_DDB is non-zero, then equal to that,
02C2 859 otherwise, last DDB in DEVLIST
02C2 860 RO = 0 - error
02C2 861 = 1 - success
02C2 862
02C2 863 Must be called at IPL=0 and KERNEL mode
02C2 864
02C2 865 .ENABL LSB
02C2 866
02C2 867 SGN$GET_DEVICE:: ; Entry with IODB MUTEX & raised IPL
02C2 868
007C 8F BB 02C2 869 PUSHR #*M<R2,R3,R4,R5,R6> ; ADDS 20 to offset to input
54 00000000'GF D0 02C6 870 ;
32 10 02C6 871 MOVL G^CTL$GL_PCB,R4 ; PICK UP PCB POINTER
02CD 872 BSBB 10$ ; Call real routine
02CF 873
007C 8F BA 02CF 874 POPR #*M<R2,R3,R4,R5,R6> ; restore regs
05 02D3 875 RSB ; Return
02D4 876
02D4 877 SGN$GET_DEVICE_LOCK_IODB: ; Entry without IODB MUTEX and IPL 0
02D4 878
007C 8F BB 02D4 879 PUSHR #*M<R2,R3,R4,R5,R6> ; ADDS 20 to offset to input
54 00000000'GF D0 02D8 880 ;
00000000'GF 16 02D8 881 MOVL G^CTL$GL_PCB,R4 ; PICK UP PCB POINTER
1A 10 02DF 882 JSB G^SCH$IOLOCKR ; GET THE IODB MUTEX FOR READ & RAISE IPL
50 DD 02E5 883 BSBB 10$ ;
54 00000000'GF D0 02E7 884 PUSHL RO ; SAVE RETURN STATUS
00000000'GF 16 02E9 885 MOVL G^CTL$GL_PCB,R4 ; PICK UP PCB POINTER
02F0 886 JSB G^SCH$IOONLOCK ; RELEASE THE IODB MUTEX
02F6 887 SETIPL #0 ; LOWER IPL
02F9 888
007C 50 8ED0 02F9 889 POPL RO ; RESTORE RETURN STATUS FROM LOCAL ROUTINE
8F BA 02FC 890 POPR #*M<R2,R3,R4,R5,R6>
05 0300 891 RSB
0301 892
2400'CF D4 0301 893 10$: CLRL W^ACF$GL_DDB ; INIT TO ZERO
2404'CF D4 0305 894 CLRL W^ACF$GL_UCB ; INIT TO ZERO
2408'CF D4 0309 895 CLRL W^ACF$GL_IDB ; INIT TO ZERO
240C'CF D4 030D 896 CLRL W^ACF$GL_CRB ; INIT TO ZERO
2418'CF D4 0311 897 CLRL W^ACF$GL_SB ; INIT TO ZERO
0315 898
56 20 AE D0 0315 899 SAVIPL -(SP) ; SAVE THE CURRENT IPL
0318 900 MOVL 32(SP),R6 ; GET ADDR OF DEVICE NAME

```

55	86	9A	031C	901	MOVZBL	(R6)+,R5	:GET SIZE OF DEVICE NAME			
7E	55	7D	031F	902	MOVQ	R5,-(SP)	:FORM DESCRIPTOR			
51	5E	D0	0322	903	MOVL	SP,R1	:ADDRESS OF DESCRIPTOR			
00000000	GF	16	0325	904	JSB	G*10C\$SEARCHALL	:SEARCH FOR DEVICE			
	8E	7C	032B	905	CLRQ	(SP)+	:GET RID OF TRASH			
			032D	906	SETIPL	(SP)+	:RESTORE OLD IPL			
2418	CF	53	D0	0330	MOVL	R3,W*ACF\$GL_SB	:STUFF THE SYSTEM BLOCK			
		04	12	0335	BNEQ	20\$	:NO ERROR, CONTINUE			
		50	D4	0337	CLRL	R0	:INDICATE ERROR			
		5E	11	0339	BRB	70\$	:EXIT			
			033B	911						
	0B	50	E8	033B	912	20\$:	BLBS	R0,25\$	:SUCCESS - FOUND DEVICE	
0908	8F	50	B1	033E	913		CMPW	R0,#SS\$NOSUCHDEV	:CHECK IF ERROR WAS 'UNIT NOT FOUND'	
		36	12	0343	914		BNEQ	60\$	:IF NOT, PUNT	
		51	D5	0345	915		TSTL	R1	:SEE IF WE GOT BACK A UCB ADDRESS	
		32	12	0347	916		BNEQ	60\$	:IF NON-ZERO, IS LISTHEAD - NO DDB FOUND	
00002400	EF	52	D0	0349	917	25\$:	MOVL	R2,L*ACF\$GL_DDB	:ADDRESS OF DDB	
54	04	A2	D0	0350	918		MOVL	DDB\$L_UCB(R2),R4	:GET ADDRESS OF FIRST UCB	
		25	13	0354	919		BEQL	60\$	:IF NO UCB, EXIT WITH OTHER FIELDS ZERO	
51	24	A4	D0	0356	920		MOVL	UCB\$L_CRB(R4),R1	:GET ADDR OF CRB	
0000240C	EF	51	D0	035A	921		MOVL	R1,L*ACF\$GL_CRB	:SAVE	
2408	CF	2C	A1	D0	0361	922	MOVL	CRB\$L_INTD+VEC\$L_IDB(R1),W*ACF\$GL_IDB	:GET ADDR OF IDB	
				0367	923					
54	A4	20	AE	B1	0367	924	30\$:	CMPW	32(SP),UCB\$W_UNIT(R4)	:IS UCB ALREADY LOADED?
		08	13	036C	925		BEQL	50\$	:BRANCH IF IT IS	
54	30	A4	D0	036E	926	40\$:	MOVL	UCB\$L_LINK(R4),R4	:GET ADDR OF NEXT UCB	
		F3	12	0372	927		BNEQ	30\$	:BR IF THERE IS ONE	
		05	11	0374	928		BRB	60\$	:EXIT WITH UCB = 0	
				0376	929					
2404	CF	54	D0	0376	930	50\$:	MOVL	R4,W*ACF\$GL_UCB		
00002410	EF	52	D0	037B	931	60\$:	MOVL	R2,ACF\$GL_LASTDDB	:LAST DDB IN LIST AS SEARCHED	
50	00000000	GF	DE	0382	932		MOVAL	G*SCS\$GA_LOCALSB,R0	:GET ADDRESS OF LOCAL SYSTEM BLOCK	
		50	D1	0389	933		CMPPL	R3,R0	:IS THIS SB LOCAL?	
		08	13	038C	934		BEQL	65\$	:YES, LEAVE NOW	
		18	A3	7D	038E	935	MOVQ	SB\$B_SYSTEMID(R3),-		
0000244C	EF			0391	936			L*BOO\$GQ_CONSYSID	:NO, SET IN THE SYSTEM ID	
				0396	937					
50	01	D0		0396	938	65\$:	MOVL	#1,R0	:SUCCESS	
		05		0399	939	70\$:	RSB			
				039A	940					
				039A	941					
				039A	942		.DSABL	LSB		



```

039A 944 .SBTTL Reset routines BOO$RESETLIST and BOO$CONRESET and BOO$MSCP_RESET
039A 945
039A 946 : BOO$CONRESET - Reset values for connect command
039A 947 :
039A 948
00000000 949 .PSECT PAGED_CODE rd,nourt,exe,long
0000 950
0000 951 .Entry BOO$CONRESET, *M<> ; Null entry mask
0002 952
0000245C'EF 00000200'EF 9E 0002 953 MOVAB L*BOO$AB PRMBUF,BOO$GL_NEXTSTR ; Reset for string allocation
00002428'EF 01 CE 000D 954 MNEGL #1,BOO$GL_CONCREG ; Null control register
0000243C'EF 01 CE 0014 955 MNEGL #1,BOO$GL_CONAUNIT ; Null adapter unit
00002434'EF 01 CE 001B 956 MNEGL #1,BOO$GL_CONVECT ; Null vector
00002438'EF 01 D0 0022 957 MOVL #1,BOO$GL_CONNUMV ; Default number of vectors
00002424'EF 02 CE 0029 958 MNEGL #2,BOO$GL_CONADP ; Invalidate adapter IR value
00002440'EF D4 0030 959 CLRL BOO$GL_CONDEV ; Clear device name pointer
00002444'EF D4 0036 960 CLRL BOO$GL_CONDRV ; and driver name pointer
00002448'EF D4 003C 961 CLRL BOO$GL_CONUNITS ; and maximum units
0000244C'EF 7C 0042 962 CLRL BOO$GL_CONSYSID ; and system id
00002458'EF D4 0048 963 CLRL BOO$GL_CONFLAGS ; and flags
00002430'EF 01 D0 004E 964 MOVL #1,L*BOO$GL_CONNUMU ; Set number of units to 1
0000241C'EF D4 0055 965 CLRL BOO$GL_COMBO_VECTOR_OFFSET ; Set vector offset from combo vectors to
00002420'EF D4 005B 966 CLRL BOO$GL_COMBO_CSR_OFFSET ; Set CSR offset from combo CSR to 0
04 0061 967 RET ; Return
0062 968
0062 969 : BOO$RESETLIST - Reset select list values
0062 970
0000 0062 971 .Entry BOO$RESETLIST, *M<> ; Null entry mask
0064 972
0000245C'EF 00002460'EF D4 0064 973 CLRL BOO$GL_SELECT ; Zap select list pointer
00000200'EF 9E 006A 974 MOVAB BOO$AB_PRMBUF,BOO$GL_NEXTSTR ; Set next string address
00002614'EF D4 0075 975 CLRL BOO$GL_SAVE_DEVNAME ; Clear autolog string
00002610'EF D4 007B 976 CLRL OUTLEN ; Clear autolog output size
0000262C'EF 00002630'EF DE 0081 977 MOVAL OUTBUF_STR,OUTBUF+4 ; Set address in descriptor of block
00002497'EF 94 008C 978 CLRB BOO$AL_ACF+ACF$B_AFLAG ; Clear ACF flags
00002430'EF 01 D0 0092 979 MOVL #1,L*BOO$GL_CONNUMU ; Set number of units to 1
04 0099 980 RET ; and return
009A 981
009A 982 : BOO$MSCP_RESET - Reset the MSCP server initialization argument list
009A 983
003C 009A 984 .Entry BOO$MSCP_RESET, *M<R2,R3,R4,R5> ; Entry mask
009C 985
FF5F CF 00 FB 009C 987 CALLS #0,BOO$CONRESET ; Reset the connect command globals
50 0084 8F 3C 00A1 988 MOVZWL #SS$ DEVOFFLINE,R0 ; Assume error
00000000'GF D5 00A6 989 TSTL G*SCS$GL_CDL ; SCS loaded?
2C 13 00AC 990 BEQL 10$ ; If eql no, error
50 02C4 8F 3C 00AE 991 MOVZWL #SS$ DEVACTIONE,R0 ; Assume error
50 03 00 02 F0 00B3 992 INSV #2,#0,#3,R0 ; Set E class error status
00000000'GF D5 00B8 993 TSTL G*SCS$GL_MSCP ; If neq already loaded
1A 12 00BE 994 BNEQ 10$ ; Exit with error
00002444'GF 000025EC'EF DE 00C0 995 MOVAL MSCP_NAME,G*BOO$GL_CONDRV ; Set pointer to MSCP server name
30 28 00CB 996 MOVCL #MSCP_ARG_LIST_SIZE, - ; Set up default argument list for
000025BC'GF 0000258C'EF 00CD 997 MSCP_ARG_LIST,G*BOO$GL_LOAD_ARGS ; MSCP server init routine
50 01 D0 00D7 998 MOVL #1,R0 ; Set success
04 00DA 999 RET ; and return
00DB 1000

```

```

00DB 1001 :
00DB 1002 : BOO$MSCP_ARG - Load MSCP arguments
00DB 1003 :
0000 00DB 1004 .Entry BOO$MSCP_ARG, ^M<> ; Entry mask
00DD 1005 :
50 20 AC D0 00DD 1006 MOVL TPA$PARAM(AP),R0 ; Get longword offset
1C AC D0 00E1 1007 MOVL TPA$NUMBER(AP),- ; Load argument value
000025BC'GF40 00E4 1008 G^BOO$GL_LOAD_ARGS[R0] ;
50 01 D0 00EA 1009 MOVL #1,R0 ; Set success
04 00ED 1010 RET ; and return
00EE 1011 :
00EE 1012 :
00EE 1013 :
00EE 1014 : BOO$MAKLIST - Make a select list entry
00EE 1015 :
007C 00EE 1016 .Entry BOO$MAKLIST, ^M<R2,R3,R4,R5,R6> ; Entry mask
00F0 1017 :
56 0000245C'EF D0 00F0 1018 MOVL L^BOO$GL_NEXTSTR,R6 ; Get pointer to next available string space
00002460'EF D5 00F7 1019 TSTL L^BOO$GL_SELECT ; Is selection pointer already set
07 12 00FD 1020 BNEQ 10$ ; Yes, continue to add entry
00002460'EF 56 D0 00FF 1021 MOVL R6,L^BOO$GL_SELECT ; Else set pointer to first select entry
50 10 AC D0 0106 1022 10$: MOVL TPA$TOKENCNT(AP),R0 ; Get string length
86 50 90 010A 1023 MOVB R0,(R6)+ ; Set count for string
66 14 BC 50 28 010D 1024 MOVC3 R0,@TPA$TOKENPTR(AP),(R6) ; Copy string body
63 94 0112 1025 CLRB (R3) ; Mark end of list
0000245C'EF 53 D0 0114 1026 MOVL R3,L^BOO$GL_NEXTSTR ; Save next string address
50 01 D0 011B 1027 MOVL #1,R0 ; Set success status
04 011E 1028 RET ;

```

```

011F 1030 .SBTTL BOO$CONADP - Set connect adapter number
011F 1031
00002424'EF 1C AC 0000 011F 1032 .Entry BOO$CONADP, ^M<>
DO 0121 1033 MOVL TPASL_NUMBER(AP),L^BOO$GL_CONADP ; Set adapter number
04 0129 1034 RET ; and return
012A 1035
00002424'EF 01 0000 012A 1036 .Entry BOO$CONNLADP ^M<> ; Connect with null adapter
CE 012C 1037 MNEGL #1,L^BOO$GL_CONADP ; Clear adapter number
04 0133 1038 RET ; and return
0134 1039
0000 0134 1040 .Entry BOO$CONVECOFFSET, ^M<> ; Offset from start of combo vectors
DO 0136 1041 MOVL TPASL_NUMBER(AP),L^BOO$GL_COMBO_VECTOR_OFFSET ; Set offset value
0000241C'EF 04 0139 1042 RET ; and return
013E 1043
0000 013F 1044 .Entry BOO$CONCSROFFSET, ^M<> ; Offset from start of combo CSRs
DO 0141 1045 MOVL TPASL_NUMBER(AP),L^BOO$GL_COMBO_CSR_OFFSET ; Set offset value
00002420'EF 04 0149 1048 RET ; and return
014A 1049
00002428'EF 1C AC 0D 00 0000 014A 1050 .Entry BOO$CONCREG, ^M<> ; Control register address
EF 014C 1051 EXTZV #0,#13,TPASL_NUMBER(AP),L^BOO$GL_CONCREG ; Set control register
04 0156 1052 RET ; and return
0157 1053
0000 0157 1054 .Entry BOO$CONVEC, ^M<> ; Set controller vector
1C AC FFFFFFFE03 8F CB 0159 1055 BICL3 #^XFFFFFFE03,TPASL_NUMBER(AP),L^BOO$GL_CONVECT ; Set vector offset
00002434'EF 04 0166 1056 RET ; and return
0167 1057
00002438'EF 1C AC 0000 0167 1058 .Entry BOO$CONCNUM, ^M<> ; Number of vectors
DO 0169 1059 MOVL TPASL_NUMBER(AP),L^BOO$GL_CONCNUMV ; Set number of vectors
04 0171 1060 RET ; and return
0172 1061
0000243C'EF 1C AC 0000 0172 1062 .Entry BOO$CONAUNIT, ^M<> ; Adapter unit number
DO 0174 1063 MOVL TPASL_NUMBER(AP),L^BOO$GL_CONAUNIT ; Set adapter unit number
04 017C 1064 RET ; and return
017D 1065
007C 017D 1066 .Entry BOO$CONDRVNAM, ^M<R2,R3,R4,R5,R6> ; Entry mask (R2-R6)
017F 1067
56 0000245C'EF DO 017F 1068 MOVL L^BOO$GL_NEXTSTR,R6 ; Address of next string storage
00002444'EF 56 DO 0186 1069 MOVL R6,BOO$GL_CONDRV ; Save pointer to driver name
86 10 AC 90 018D 1070 MOVB TPASL_TOKENCNT(AP),(R6)+ ; Set count for string
0000245C'EF 56 10 AC C1 0191 1071 ADDL3 TPASL_TOKENCNT(AP),R6,BOO$GL_NEXTSTR ; Mark string allocated
66 14 BC 10 AC 28 019A 1072 MOVC3 TPASL_TOKENCNT(AP),@TPASL_TOKENPTR(AP),(R6) ; Copy string
50 01 DO 01A0 1073 MOVL #1,R0 ; and return success
04 01A3 1074 RET
01A4 1075
00FC 01A4 1076 .Entry BOO$DEVNAME, ^M<R2,R3,R4,R5,R6,R7> ; Device name/unit
01A6 1077
56 0000245C'EF DO 01A6 1078 MOVL BOO$GL_NEXTSTR,R6 ; Get pointer to next available string
54 14 AC DO 01AD 1079 MOVL TPASL_TOKENPTR(AP),R4 ; Get pointer to string
53 10 AC DO 01B1 1080 MOVL TPASL_TOKENCNT(AP),R3 ; And number of characters
00002588'EF D4 01B5 1081 CLRL FULL_NAME_PTR ; Initialize full device name
57 86 9E 01BB 1082 MOVAB (R6)+,R7 ; Save pointer
64 53 24 3A 01BE 1083 LOCC #^A/$/,R3,(R4) ; Find any possible '$'
22 13 01C2 1084 BEQL B$ ; None, just continue
00002588'EF 57 DO 01C4 1085 MOVL R7,FULL_NAME_PTR ; Store pointer

```



55	53	50	C3	01CB	1086	SUBL3	R0,R3,R5	: Number of characters in node
67	55	01	81	01CF	1087	ADDB3	#1,R5,(R7)	: Set in size (incl '\$')
		03	BB	01D3	1088	PUSHR	#*M<R0,R1>	: Save registers
66	64	53	28	01D5	1089	MOVC3	R3,(R4),(R6)	: Copy full string
	56	53	D0	01D9	1090	MOVL	R3,R6	: Save ending address
		03	BA	01DC	1091	POPR	#*M<R0,R1>	: Restore registers
53	50	01	C3	01DE	1092	SUBL3	#1,R0,R3	: Number of characters left
54	51	01	C1	01E2	1093	ADDL3	#1,R1,R4	: Pointer to string
	55	86	9E	01E6	1094	MOVAB	(R6)+,R5	: Save pointer to count byte
		65	94	01E9	1095	CLRB	(R5)	: Initialize count to zero
		52	D4	01EB	1096	CLRL	R2	: Initialize unit accumulator
	50	84	9A	01ED	1097	MOVZBL	(R4)+,R0	: Get a character from device name
	30	50	91	01F0	1098	CMPB	R0,#*A/0/	: And check for a digit
		05	1F	01F3	1099	BLSSU	20\$	: Branch if not
	39	50	91	01F5	1100	CMPB	R0,#*A/9/	: Final check for digit
		0F	1B	01F8	1101	BLEQU	40\$	: Yes it is
	86	50	90	01FA	1102	MOVB	R0,(R6)+	: Part of device name
		65	96	01FD	1103	INCB	(R5)	: Increase count
		67	96	01FF	1104	INCB	(R7)	: Including nodename
	E9	53	F5	0201	1105	SOBGTR	R3,10\$	: Continue
		16	11	0204	1106	BRB	50\$	
	50	84	9A	0206	1107	MOVZBL	(R4)+,R0	: Get another digit
	50	30	C2	0209	1108	SUBL	#*A/0/,R0	: Get value
	52	0A	C4	020C	1109	MULL	#10,R2	: Scale accumulator before adding digit
		2F	19	020F	1110	BLSS	60\$	: Error
	50	09	D1	0211	1111	CMPL	#9,R0	: Check for numeric
		2A	19	0214	1112	BLSS	60\$	: Error if not
	52	50	C0	0216	1113	ADDL	R0,R2	: And add new digit
	EA	53	F5	0219	1114	SOBGTR	R3,30\$	: Continue for entire unit number
0000245C'EF		56	D0	021C	1115	MOVL	R6,BOOSGL_NEXTSTR	: Save updated string pointer
0000242C'EF		52	D0	0223	1116	MOVL	R2,BOOSGL_CONCUNIT	: Set unit number
0000243C'EF		52	D0	022A	1117	MOVL	R2,BOOSGL_CONAUNIT	: Assume same for adapter unit
00002440'EF		55	D0	0231	1118	MOVL	R5,BOOSGL_CONDEV	: Save device name pointer
		65	95	0238	1119	TSTB	(R5)	: Must not be null device name
		04	13	023A	1120	BEQL	60\$	: Error if so
	50	01	D0	023C	1121	MOVL	#1,R0	: Return success
			04	023F	1122	RET		: and return
		50	D4	0240	1123	CLRL	R0	: Return error status
			04	0242	1124	RET		
				0243	1125			
00002448'EF	1C	AC	0000	0243	1126	.Entry	BOOSCONUNITS, *M<>	: Maximum units to be connected
			D0	0245	1127	MOVL	TPASL_NUMBER(AP),L*BOOSGL	: CONUNITS ; Set maximum units
			04	024D	1128	RET		: and return
				024E	1129			
0000244C'EF	1C	AC	0000	024E	1130	.Entry	BOOSCONSYSID LOW, *M<>	: System ID
			D0	0250	1131	MOVL	TPASC_NUMBER(AP), -	
			04	0258	1132		L*BOOSGQ_CONSYSID	: Set System ID (low longword)
				0258	1133	RET		: and return
				0259	1134			
00002450'EF	1C	AC	0000	0259	1135	.Entry	BOOSCONSYSID HIGH, *M<>	: System ID
			D0	025B	1136	MOVL	TPASC_NUMBER(AP), -	
			04	0263	1137		L*BOOSGQ_CONSYSID+4	: Set System ID (high longword)
				0263	1138	RET		: and return
				0264	1139			
			0000	0264	1140	.Entry	BOOSCONSOLE, *M<>	: Connect console block stor. device
00002424'EF	01	CE		0266	1141	MNEGL	#1,L*BOOSGL_CONADP	: No adapter
				0266	1142			

CONFIGUTL  
V04-002

1 3  
- SYSGEN UTILITIES FOR CONFIGURE PROCESS 15-SEP-1984 23:46:56 VAX/VMS Macro V04-00  
BOO\$CONADP - Set connect adapter number 14-SEP-1984 16:09:11 [BOOTS.SRC]SYSGEN.MAR;3

Page 25  
(4)

0000243C'EF	01	D0	026D	1143	MOVL	#1,L^BOO\$GL_CONAUNIT	: Set adapter unit = 1 (not used)
0000242C'EF	01	D0	0274	1144	MOVL	#1,L^BOO\$GL_CONCUNIT	: Set unit = 1
00002440'EF	000024D6'EF	9E	027B	1145	MOVAB	L^CONSNAME,L^BOO\$GL_CONDEV	: Set device name pointer
00002438'EF	02	D0	0286	1146	MOVL	#2,L^BOO\$GL_CONNUMV	: Set 2 vectors
00002428'EF		D4	028D	1147	CLRL	L^BOO\$GL_CONCREG	: No control register
00002430'EF	01	D0	0293	1148	MOVL	#1,L^BOO\$GL_CONNUMU	: Set number of units to 1
00002448'EF	02	D0	029A	1149	MOVL	#2,L^BOO\$GL_CONUNITS	: Set max units to 2 (OPA0 is 1st unit)
00000000'EF		16	02A1	1150	JSB	IOGEN\$CONSOLE	: Do cpu dependent stuff
50	01	D0	02A7	1151	MOVL	#1,R0	
		04	02AA	1152	RET		

```

02AB 1154 .SBTTL BOOS$CONNECT - Connect specified device and load driver
02AB 1155 :
02AB 1156 : BOOS$CONNECT - Allows a single device to be introduced, appropriate data
02AB 1157 : structures allocated and initialized, the driver loaded if
02AB 1158 : required and the controller and device initialized.
02AB 1159 :
OFFC 02AB 1160 .Entry BOOS$CONNECT, ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;
02AD 1161 :
FD50' 30 02AD 1162 BSBW BOOS$LOCK_GEN ; Lock SYSGEN database
7F 50 E9 02B0 1163 BLBC R0,70$ ; If error, exit
02B3 1164 :
02B3 1165 : Value of BOOS$GL_CONADP
02B3 1166 :
02B3 1167 : 0 or greater => /ADAPTER=n specified
02B3 1168 : -1 => /NOADAPTER specified
02B3 1169 : -2 => not specified
02B3 1170 :
02B3 1171 :
02B3 1172 :
00002424'EF D5 02B3 1173 5$: TSTL L^BOOS$GL_CONADP ; Has an adapter been specified?
44 18 02B9 1174 BGEQ 20$ ; If so, branch
00002424'EF FFFFFFFF 8F D1 02BB 1175 CMPL #-1,L^BOOS$GL_CONADP ; Null adapter?
28 13 02C6 1176 BEQL 10$ ; Branch if yes
00002424'EF FFFFFFFE 8F D1 02CB 1177 CMPL #-2,L^BOOS$GL_CONADP ; None specified in CONNECT?
09 13 02D3 1178 BEQL 7$ ; Figure it out from the database
50 007C80D2 8F D0 02D5 1179 MOVL #SYSG$_NOADAPTER,R0 ; Set no adapter specified error
45 11 02DC 1180 BRB 60$ ; exit
02DE 1181 :
35 50 E9 02DE 1182 7$: $CMKRNLS W^CONN_ADAP ; Get adapter number from I/O database
C3 11 02EB 1183 BLBC -R0,60$ ; Exit with error
02EE 1184 BRB 5$ ; Dispatch now on adapter type
02F0 1185 :
OD 11 02F0 1186 10$: $CMKRNLS W^CONNADP ; Change mode to see data base
02FD 1187 BRB -30$ ; Continue
02FF 1188 :
0E 50 E9 02FF 1189 20$: $CMKRNLS W^CONNECT ; Change mode to see data base
00000000'EF 0000248C'EF FA 030C 1190 30$: BLBC -R0,40$ ; Error occurred
03 50 E8 030F 1191 CALLG L^BOOS$AL_ACF,IOGEN$LOADER ; Load database and driver
FCE0' 30 031D 1193 40$: BSBW PUTERROR ; Branch if success
50 01 D0 0320 1194 50$: MOVL #1,R0 ; Give error message
50 DD 0323 1195 60$: PUSHL R0 ; Set success for parser
FCD8' 30 0325 1196 BSBW BOOS$UNLOCK_GEN ; Save error status
03 50 E8 0328 1197 BLBS R0,65$ ; Unlock SYSGEN database
FCD2' 30 032B 1198 BSBW PUTERROR ; If no error, continue
50 8ED0 032E 1199 65$: POPL R0 ; Give error message
04 0331 1200 RET ; Restore status
FCCB' 30 0332 1201 70$: BSBW PUTERROR ; Give error message
04 0335 1202 RET ;
0336 1203 :
0336 1204 : Local routine to get adapter number from I/O database
0336 1205 : Must be called by a CMKRNLS since SGN$GET_DEVICE must be called
0336 1206 : in Kernel mode.
0336 1207 :
0000 0336 1208 .Entry CONN_ADAP, ^M<>
0338 1209 :
7E 0000242C'EF 3C 0338 1210 MOVZWL L^BOOS$GL_CONCUNIT,-(SP) ; Unit number

```



00002440'EF	DD	033F	1211	PUSHL	L^BOOS\$GL CONDEV	; Device name
000002D4'EF	16	0345	1212	JSB	SGNS\$GET_DEVICE_LOCK_IDDB	; Get device data base addresses
SE 08	CO	0348	1213	ADDL2	#8,SP	; Pop off input parameters
		034E	1214			
50 00002408'EF	DO	034E	1215	MOVL	L^ACF\$GL_IDB,R0	; Address of IDB
	09	12	0355	BNEQ	5\$	; Error if zero
50 007C80D2 8F	DO	0357	1217	MOVL	#SYSG\$_NOADAPTER,R0	; Set no adapter specified error
	18	11	035E	BRB	20\$	; Branch to exit
		0360	1219			
00002424'EF	01	CE	0360	1220	5\$: MNEGL	#1,L^BOOS\$GL CONADP ; Assume null adapter
50 14 A0	DO	0367	1221	MOVL	IDB\$L_ADP(R0),R0	; Address of ADP block
	08	13	036B	1222	BEQL	10\$ ; Null adapter if zero
00002424'EF	0C A0	3C	036D	1223	MOVZWL	ADPSW_TR(R0),L^BOOS\$GL_CONADP ; Set adapter number
		0375	1224			
50 01	DO	0375	1225	10\$: MOVL	#1,R0	; Set success
	04	0378	1226	20\$: RET		; Return
		0379	1227			

```

0379 1229 .ENABL LSB
0379 1230 ; Connect with null adapter
0379 1231
OFFC 0379 1232 .Entry CONNLADP, *M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
037B 1233
50 00002588'EF D0 037B 1234 MOVL L^FULL_NAME_PTR,R0 ; Try full device name
07 12 0382 1235 BNEQ 5$ ; Good, continue
50 00002440'EF D0 0384 1236 MOVL L^BOO$GL_CONDEV,R0 ; Use normal name
5A 0000248C'EF 9E 038B 1237 5$: MOVAB L^BOO$AL_ACF,R10 ; Address ACF
6A D4 0392 1238 CLRL ACF$S_ADAPTER(R10) ; Set no adapter
04 AA D4 0394 1239 CLRL ACF$S_CONFIGREG(R10) ; Set address of config reg
08 AA B4 0397 1240 CLRW ACF$S_AVECTOR(R10) ; Set SCB offset for adapter
01 E1 039A 1241 BBC #ACF$V_CRBBLT,- ; Br. if CRB built flag is clear
4C 00002458'EF 039C 1242 BOO$GL_CONFLAGS,17$
6A 00002454'EF D0 03A2 1243 MOVL BOO$GL_CONCRB,ACF$S_ADAPTER(R10) ; Store CRB address
43 11 03A9 1244 BRB 17$ ; Join common code
03AB 1245
OFFC 03AB 1246 .Entry CONNECT, *M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;
03AD 1247
5B FFFFFFFC'GF 9E 03AD 1248 MOVAB G^IOC$GL_ADPLIST-ADP$S_LINK,R11 ; Get address of adapter list
5B 04 AB D0 03B4 1249 10$: MOVL ADP$S_LINK(R11),R11 ; Flink onward through adapter list
08 12 03B8 1250 BNEQ 15$ ; Continue if another adapter
50 007C80BA 8F D0 03BA 1251 MOVL #SYSG$_INVADAP,R0 ; Set invalid adapter error
04 03C1 1252 RET ; Return
03C2 1253
00002424'EF 0C AB B1 03C2 1254 15$: CMPW ADP$S_TR(R11),L^BOO$GL_CONADP ; Is this the specified TR?
E8 12 03CA 1255 BNEQ 10$ ; No, try another
5A 0000248C'EF 9E 03CC 1256 MOVAB L^BOO$AL_ACF,R10 ; Get address of ACF
6A 5B D0 03D3 1257 MOVL R11,ACF$S_ADAPTER(R10) ; Set address of ADP
04 AA 6B D0 03D6 1258 MOVL ADP$S_CSR(R11),ACF$S_CONFIGREG(R10) ; Set address of config reg
50 1C AB 00000000'GF C3 03DA 1259 SUBL3 G^EXE$GL_SCB,- ; Calculate offset into SCB of
08 AA 50 B0 03E3 1260 ADP$S_AVECTOR(R11),R0 ; adapter's interrupt vectors.
50 00002440'EF D0 03E7 1261 MOVW R0,ACF$S_AVECTOR(R10) ; Store offset in ACF.
03EE 1262 MOVL L^BOO$GL_CONDEV,R0 ; Device name
03EE 1263
14 AA 00002440'EF D0 03EE 1264 17$: MOVL BOO$GL_CONDEV,ACF$S_DEVNAME(R10); Set pointer to device name
03F6 1265
03F6 1266 ; Now try to get driver name from DDB if it exists and load BOO$GL_CONSYSID
03F6 1267 ; if HSC device.
03F6 1268
7E 0000242C'EF 3C 03F6 1269 MOVZWL L^BOO$GL_CONCUNIT,-(SP) ; Unit number
50 DD 03FD 1271 PUSHL R0 ; Device name
000002D4'EF 16 03FF 1272 JSB SGN$GET_DEVICE_LOCK_IODB; Get device data base addresses
5E 08 C0 0405 1273 ADDL2 #8,SP ; Pop off input parameters
08 50 E8 0408 1274 BLBS R0,20$ ; All okay
50 007C9010 8F D0 040B 1275 MOVL #SYSG$_NODEV,R0 ; Set error code - 'Device not known'
04 0412 1276 RET ; Leave
0413 1277
05 E2 0413 1278 20$: BBSS #ACF$V_GETDONE,-
00 00002458'EF 0415 1279 L^BOO$GL_CONFLAGS,21$ ; Notify LOADER that GET was done
041B 1280
18 AA 00002444'EF D0 041B 1281 21$: MOVL BOO$GL_CONDRV,ACF$S_DRVNAME(R10); And driver name
31 14 0423 1282 BGTR 30$ ; Branch if driver specified
51 00002400'EF D0 0425 1283 MOVL ACF$GL_DDB,R1 ; DDB address
07 13 042C 1284 BEQL 25$ ; Branch if none
18 AA 24 A1 DE 042E 1285 MOVAL DDB$T_DRVNAME(R1),ACF$S_DRVNAME(R10) ; Address from DDB

```

```

21 11 0433 1286 BRB 30$ : Branch around name hackery
      0435 1287
56 0000245C'EF D0 0435 1288 25$: MOVL L*BOOS$GL_NEXTSTR,R6 : Get address of next free space
      18 AA 56 D0 043C 1289 : MOVL R6,ACF$S_L_DRVNAME(R10) : Set as driver name address
      86 08 90 0440 1290 : MOVW #8,(R6)+ : Set count for string
66 52455649 52442020 8F 7D 0443 1291 : MOVQ #^A/ DRIVER/,(R6) : Set driver suffix
      51 14 AA D0 044E 1292 : MOVL ACF$S_L_DEVNAME(R10),R1 : Pointer to device name
      66 01 A1 B0 0452 1293 : MOVW 1(R1),(R6) : Form default driver name
      0456 1294
      0A AA 0000243C'EF 90 0456 1295 30$: MOVW BOOS$GL_CONAUNIT,ACF$S_L_AUNIT(R10); Set adapter unit
      21 AA 00002430'EF 90 045E 1296 : MOVW L*BOOS$GL_CONNUMU,ACF$S_L_NUMUNIT(R10)
      0466 1297 : Store number of units to configure
      0B AA 00002438'EF 90 0466 1298 : MOVW BOOS$GL_CONFLAGS,ACF$S_L_AFLAG(R10) : Store flags
0000241C'EF 00002434'EF A1 046E 1299 : ADDW3 BOOS$GL_CONVECT,BOOS$GL_COMBO_VECTOR_OFFSET,-;
      10 AA 0479 1300 : ACF$S_L_VECTOR(R10) : Set vector address
50 0000241C'EF 08 02 EF 047B 1301 : EXTZV #2,#8,BOOS$GL_COMBO_VECTOR_OFFSET,R0; Save vector offset in longwords
      1F AA 50 90 0484 1302 : MOVW R0,ACF$S_L_COMBO_VECTOR_OFFSET(R10);
      0488 1303 :
      0488 1304 : Set up ACF$S_L_CONTRLREG - can either be UNIBUS CSR or address of CI
      0488 1305 : System id.
      0488 1306 :
      0000244C'EF D5 0488 1307 : TSTL BOOS$GL_CONSYSID : See if SYSIDLOW was specified
      0A 13 048E 1308 : BEQL 40$ : Branch if not
      0C AA 0000244C'EF 9E 0490 1309 : MOVAB BOOS$GL_CONSYSID, -
      0498 1310 : ACF$S_L_CONTRLREG(R10) : Set system id address
      22 11 0498 1311 : BRB 50$ : Branch
      049A 1312 :
      049A 1313 : Calculate system virtual address of UNIBUS CSR
      049A 1314 :
      00002428'EF 00001000 8F C1 049A 1315 40$: ADDL3 #UBA_IOBASE, -
      0C AA 04A5 1316 : BOOS$GL_CONCREG, -
      04A7 1317 : ACF$S_L_CONTRLREG(R10) : control register address
      0C AA 04 AA C0 04A7 1318 : ADDL ACF$S_L_CONFIGREG(R10), -
      04AC 1319 : ACF$S_L_CONTRLREG(R10) : Add adapter va base
      00002420'EF C0 04AC 1320 : ADDL BOOS$GL_COMBO_CSR_OFFSET,-; Add offset to get true CSR address
      0C AA 04B2 1321 : ACF$S_L_CONTRLREG(R10)
      00002420'EF 8E 04B4 1322 : MNEGB BOOS$GL_COMBO_CSR_OFFSET,-; Calculate offset back to CSR start
      20 AA 04BA 1323 : ACF$S_L_COMBO_CSR_OFFSET(R10); Save offset
      04BC 1324
      12 AA 0000242C'EF B0 04BC 1325 50$: MOVW BOOS$GL_CONCUNIT, -
      04C4 1326 : ACF$S_L_UNIT(R10) : Set controller unit number
      1C AA 00002448'EF B0 04C4 1327 : MOVW BOOS$GL_CONUNITS, -
      04CC 1328 : ACF$S_L_MAXUNITS(R10) : Set maximum units
      1E AA 00002438'EF 90 04CC 1329 : MOVW BOOS$GL_CONNUMV, -
      50 01 D0 04D4 1330 : MOVW ACF$S_L_NUMVEC(R10) : Set count of vectors
      04 04D7 1331 55$: MOVL #1,R0 : Set success
      04D8 1332 :
      04D8 1333 :
      04D8 1334 :
      04D8 1335 : .DSABL LSB

```



```

      04D8 1337      .SBTTL BOO$LOAD - Load a driver or misc code if not already loaded
      04D8 1338      :
      04D8 1339      : BOO$LOAD - Loads the driver or misc code if not already loaded.
      04D8 1340      :
OFFC  04D8 1341      .Entry BOO$LOAD, ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
      04DA 1342
52    D4  04DA 1343      CLRL  R2      ; Clear reload flag
05    11  04DC 1344      BRB   LOADRV   ; And merge with common code
  
```

```

04DE 1346      .SBTTL BOOS$RELOAD - Reload a specified driver
04DE 1347      :
04DE 1348      : BOOS$RELOAD - Reloads the specified driver replacing any existing copy
04DE 1349      : unless there are busy units requiring the driver that would
04DE 1350      : be replaced.
04DE 1351      :
OFFC 04DE 1352      .Entry BOOS$RELOAD, *M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ;
04E0 1353      :
52 01 90 04E0 1354      MOVB    #ACF$M_RELOAD,R2          ; Set flag to force reload
04E3 1355      LOADRV:
04E3 1356      :
04E3 1357      : The first block of the file will be read to determine if it is a driver or
04E3 1358      : misc code by looking at the type field.
04E3 1359      :
00002444'EF DD 04E3 1360      PUSHL   BOOS$GL_CONDRV          ; File name string
6E D6 04E9 1361      INCL     (SP)                          ; Get past the count
7E 00002444'FF 9A 04EB 1362      MOVZBL  @BOOS$GL_CONDRV,-(SP) ; Length of file name
57 5E D0 04F2 1363      MOVL     SP,R7                      ; Address of descriptor for file name
FB08' 30 04F5 1364      BSBW     BOOS$EXEOPEN              ; Open the file (default SYS$SYSTEM:.EXE)
5E 50 E9 04F8 1365      BLBC     R0,40$                     ; Error
5E 08 C0 04FB 1366      ADDL     #8,SP                       ; Clean up stack
56 00002200'EF 9E 04FE 1367      MOVAB   BOOS$AB_LOADBUF,R6   ; Buffer for file read
58 02 D0 0505 1368      MOVL     #2,R8                      ; First block after image header
59 01 D0 0508 1369      MOVL     #1,R9                      ; One page
FAF2' 30 050B 1370      BSBW     BOOS$READFILE              ;
48 50 E9 050E 1371      BLBC     R0,40$                     ; Error
FAEC' 30 0511 1372      BSBW     BOOS$FILCLOSE              ; Close the currently open file
42 50 E9 0514 1373      BLBC     R0,40$                     ; Error
0000220A'EF 91 0517 1374      CMPB     BOOS$AB_LOADBUF+SLV$B_TYPE,- ;
62 8F 051D 1375      #DYN$C_LOADCODE                      ; Check for misc code
3C 13 051F 1376      BEQL     LOADCODE
FADC' 30 0521 1377      BSBW     BOOS$LOCK_GEN              ; Lock SYSGEN database
32 50 E9 0524 1378      BLBC     R0,40$                     ; If lbc, didn't get lock
5A 0000248C'EF 9E 0527 1379      MOVAB   L^BOOS$AL_ACF,R10   ; Get base address for ACF block
18 AA 00002444'EF D0 052E 1380      MOVL     BOOS$GL_CONDRV,ACF$L_DRVNAME(R10) ;
0B AA 52 90 0536 1381      MOVB     R2,ACF$B_AFLAG(R10)      ; Set flags for load or reload
14 AA D4 053A 1382      CLRL     ACF$L_DEVNAME(R10)          ; No device name
00000000'EF 6A FA 053D 1383      CALLG   (R10)-L^IOGEN$LOADER ; Load requested driver
03 50 E8 0544 1384      BLBS     R0,20$                     ; Continue if no error
FAB6' 30 0547 1385      BSBW     PUTERROR                    ; Give error message
50 DD 054A 1386 20$:  PUSHL     R0                          ; Save status
FAB1' 30 054C 1387      BSBW     BOOS$UNLOCK_GEN             ; Unlock SYSGEN database
03 50 E8 054F 1388      BLBS     R0,30$                     ; If no error, continue
FAAB' 30 0552 1389      BSBW     PUTERROR                    ; Give error message
50 BED0 0555 1390 30$:  POPL     R0                          ; Restore status
FAA4' 04 0558 1391      RET                                ; Exit
04 0559 1392 40$:  BSBW     PUTERROR                    ; Give error message
04 055C 1393      RET                                ; Exit
055D 1394      :
055D 1395      LOADCODE:
00002444'EF DD 055D 1396      PUSHL   BOOS$GL_CONDRV          ; File name string
6E D6 0563 1397      INCL     (SP)                          ; Get past the count
7E 00002444'FF 9A 0565 1398      MOVZBL  @BOOS$GL_CONDRV,-(SP) ; Length of file name
57 5E D0 056C 1399      MOVL     SP,R7                      ; Address of descriptor for file name
FA8E' 30 056F 1400      BSBW     BOOS$UFOOPEN              ; Open the file for user access (default SYS
22 50 E9 0572 1401      BLBC     R0,10$                     ; Error
5E 08 C0 0575 1402      ADDL     #8,SP                       ; Clean up stack

```

```

00002200'EF 9F 0578 1403 PUSHAB BOO$AB_LOADBUF ; Use code buffer for return address array
          51 DD 057E 1404 PUSHL R1 ; Channel
          02 DD 0580 1405 PUSHL #2 ; Arg count
50 5E D0 0582 1406 MOVL SP,R0
          0585 1407 $CMKRNLS ROUTIN = EXE$LOAD_CODE,-
          0585 1408 ARGST = (R0)
04 50 E8 0594 1409 BLBS R0,20$
FA66 30 0597 1410 10$: BSBW PUTERROR
          04 059A 1411 RET
          059B 1412
          059B 1413 20$: $CMKRNLS ROUTIN = LINK_CODE
EA 50 E9 05AA 1414 BLBC R0,10$
          04 05AD 1415 RET
          05AE 1416 ;
          05AE 1417 LINK_CODE:
001C 05AE 1418 .WORD
52 00002200'GF D0 05B0 1419 MOVL G^BOO$AB_LOADBUF,R2 ; Address of loaded code
          54 52 D0 05B7 1420 MOVL R2,R4 ; Save address of loaded code
          53 10 A4 D0 05BA 1421 MOVL SLV$A SYSVECS(R4),R3 ; Get address of vectors in SYS.EXE
00000000'GF 16 05BE 1422 JSB G^EXE$LINK_VEC ; Connect vectors to loaded routines.
          10 50 E9 05C4 1423 BLBC R0,10$ ; Leave on error
5C 000025BC'GF DE 05C7 1424 MOVAL G^BOO$GL_LOAD_ARGS,AP ; Argument list for initialization routine
          50 04 A4 D0 05CE 1425 MOVL SLV$L_INITRTN(R4),R0 ; Possible initialization routine
          03 13 05D2 1426 BEQL 10$ ; None, leave
          6044 16 05D4 1427 JSB (R0)[R4] ; Call it
          04 05D7 1428 10$: RET
          05D8 1429

```



```

00000000'GF 003C 05D8 1431 .SBTTL BOO$GIVEHELP - Print Help information
00002574'EF 9F 05D8 1432 :
00002559'EF 9F 05D8 1433 : Print Help Information
08 AC B0 05D8 1434 :
00002578'EF 9F 05DA 1435 .Entry BOO$GIVEHELP, ^M<R2,R3,R4,R5> :
0C AC D0 05DA 1436 :
0000257C'EF 9F 05DA 1437 PUSHAB G^LIB$GET_INPUT : Input routine
00002578'EF 9F 05E0 1438 PUSHAB HELP_FLAG : Flags
00002578'EF 9F 05E6 1439 PUSHAB L^HELP_FILE : Library
7E D4 B0 05EC 1440 MOVW TPASL_STRINGCNT(AP),- :
00000000'GF 9F 05EF 1441 HELP_DESC : Set length
0000257C'EF 9F 05F4 1442 MOVL TPASC_STRINGPTR(AP),- :
00002578'EF 9F 05F7 1443 HELP_DESC+4 : Set address
7E D4 B0 05FC 1444 PUSHAB HELP_DESC : Input string
00000000'GF 9F 0602 1445 CLRL -(SPT) : Width
00000000'GF 9F 0604 1446 PUSHAB G^LIB$PUT_OUTPUT : Output routine
00000000'GF 06 FB 060A 1447 CALLS #6,G^LBR$OUTPUT_HELP : Call help routine
04 0611 1448 RET : Return with status
0612 1450
0612 1451 .END

```

CONFIGUTL  
Symbol table

- SYSGEN UTILITIES FOR CONFIGURE PROCESS 15-SEP-1984 23:46:56 VAX/VMS Macro V04-00  
14-SEP-1984 16:09:11 [BOOTS.SRC]SYSGEN.MAR;3

Page 34  
(4)

\$ST2	= 00000005			BOOSDEVNAME	000001A4	RG	06
\$CLI.	= 00002464	R	04	BOOSEXEOPEN	*****	X	06
\$CLI..	= 00002480	R	04	BOOSFILCLOSE	*****	X	06
ACFSB_AFLAG	= 0000000B			BOOSGB_FILELEN	000024FD	RG	04
ACFSB_AUNIT	= 0000000A			BOOSGIVEHELP	000005D8	RG	06
ACFSB_CNUMVEC	= 0000001E			BOOSGL_CMDOPT	*****	X	05
ACFSB_COMBO_CSR_OFFSET	= 00000020			BOOSGL_COMBO_CSR_OFFSET	00002420	RG	04
ACFSB_COMBO_VECTOR_OFFSET	= 0000001F			BOOSGL_COMBO_VECTOR_OFFSET	0000241C	RG	04
ACFSB_NUMUNIT	= 00000021			BOOSGL_CONADP	00002424	RG	04
ACFSC_LENGTH	= 00000028			BOOSGL_CONAUNIT	0000243C	RG	04
ACFSGC_CRB	0000240C	RG	04	BOOSGL_CONCRB	00002454	RG	04
ACFSGL_DDB	00002400	RG	04	BOOSGL_CONCREG	00002428	RG	04
ACFSGL_DPT	00002414	RG	04	BOOSGL_CONCUNIT	0000242C	RG	04
ACFSGL_IDB	00002408	RG	04	BOOSGL_CONDEV	00002440	RG	04
ACFSGL_LASTDDB	00002410	RG	04	BOOSGL_CONDRV	00002444	RG	04
ACFSGL_SB	00002418	RG	04	BOOSGL_CONFLAGS	00002458	RG	04
ACFSGL_UCB	00002404	RG	04	BOOSGL_CONNUMU	00002430	RG	04
ACFSL_ADAPTER	= 00000000			BOOSGL_CONNUMV	00002438	RG	04
ACFSL_CONFIGREG	= 00000004			BOOSGL_CONUNITS	00002448	RG	04
ACFSL_CONTRLREG	= 0000000C			BOOSGL_CONVECT	00002434	RG	04
ACFSL_DEVNAME	= 00000014			BOOSGL_FILEADDR	000024F9	RG	04
ACFSL_DRVNAME	= 00000018			BOOSGL_LOAD_ARGS	000025BC	RG	04
ACFSM_RELOAD	= 00000001			BOOSGL_NEXTSTR	0000245C	RG	04
ACFSV_CRBBLT	= 00000001			BOOSGL_PARINUSE	000024FE	RG	04
ACFSV_GETDONE	= 00000005			BOOSGL_RETSAVE	000024C4	RG	04
ACFSV_NOLOAD_DB	= 00000003			BOOSGL_SELECT	00002460	RG	04
ACFSW_AVECTOR	= 00000008			BOOSGQ_CMDESC	= 0000246C	RG	04
ACFSW_CUNIT	= 00000012			BOOSGQ_CONSYSID	0000244C	RG	04
ACFSW_CVECTOR	= 00000010			BOOSGQ_LIMITS	000024B4	RG	04
ACFSW_MAXUNITS	= 0000001C			BOOSGQ_RETADR	000024BC	RG	04
ADPSL_AVECTOR	= 0000001C			BOOSGT_ACTIVE	0000250A	RG	04
ADPSL_CSR	= 00000000			BOOSGT_CURRENT	00002502	RG	04
ADPSL_LINK	= 00000004			BOOSGT_CVNAME	000024DE	RG	04
ADPSW_TR	= 0000000C			BOOSGT_DDNAME	000024F0	RG	04
AUTOLOG	00000206	RG	05	BOOSGT_DEFAULT	00002511	RG	04
BOOSAB_LOADBUF	00002200	R	04	BOOSGT_DYNAME	000024E7	RG	04
BOOSAB_PATCH	00000000	RG	04	BOOSGT_FILE	00002519	RG	04
BOOSAB_PRMBUF	00000200	RG	04	BOOSGT_OPNAME	000024DA	RG	04
BOOSAL_ACF	0000248C	RG	04	BOOSGT_PROMPT	00002480	RG	04
BOOSAL_CLIBLK	00002464	RG	04	BOOSGT_SAVE_DEVNAME	00002614	R	04
BOOSCONADP	0000011F	RG	06	BOOSHICIM	00000000	RG	03
BOOSCONAUNIT	00000172	RG	06	BOOSLOAD	000004D8	RG	06
BOOSCONCNUM	00000167	RG	06	BOOSLOCK_GEN	*****	X	05
BOOSCONCREG	0000014A	RG	06	BOOSLOLIM	00000000	RG	02
BOOSCONCSROFFSET	0000013F	RG	06	BOOSMAKLIST	000000EE	RG	06
BOOSCONCVEC	00000157	RG	06	BOOSMSCP_ARG	000000DB	RG	06
BOOSCONDRVNAM	0000017D	RG	06	BOOSMSCP_RESET	0000009A	RG	06
BOOSCONF IGALL	00000000	RG	05	BOOSREADFILE	*****	X	06
BOOSCONF IGONE	0000006F	RG	05	BOOSRELOAD	000004DE	RG	06
BOOSCONNECT	000002AB	RG	06	BOOSRESETLIST	00000062	RG	06
BOOSCONLADP	0000012A	RG	06	BOOSUFOOPEN	*****	X	06
BOOSCONRESET	00000000	RG	06	BOOSUNLOCK_GEN	*****	X	05
BOOSCONSOLE	00000264	RG	06	BOOSUSEACT	000027D0	RG	04
BOOSCONSYSID_HIGH	00000259	RG	06	BOOCMD\$V_AUTOLOG	= 0000000C		
BOOSCONSYSID_LOW	0000024E	RG	06	BOOCMD\$V_EXCLUDE	= 00000007		
BOOSCONUNITS	00000243	RG	06	BOOCMD\$V_SELECT	= 00000006		
BOOSCONVECOFFSET	00000134	RG	06	CLISB_RQTYPE	= 00000000		

CONFIGUTL  
Symbol table

- SYSGEN UTILITIES FOR CONFIGURE PROCESS F 4  
15-SEP-1984 23:46:56 VAX/VMS Macro V04-00  
14-SEP-1984 16:09:11 [BOOTS.SRC]SYSGEN.MAR;3

Page 35  
(4)

CLISC_REQDESC	= 0000001C			OPERGETJPI	00002694	R	04
CLISK_GETCMD	= 00000001			OPERMSG	000026C0	R	04
CLISW_RQSIZE	= 00000008			OPERMSGBUF	000026C8	R	04
CONFIGADP	00000103	RG	05	OPERMSGFAO	000026AC	R	04
CONFIGSW	= 00000001			OPERMSGID	000026A8	R	04
CONFIG_EXIT	0000005D	R	05	OPERMSGNAM	000026B4	R	04
CONNECT	000003AB	RG	06	OPERMSGPID	000026B0	R	04
CONNADP	00000379	RG	06	OPERMSGTXT	000026D0	R	04
CONN ADAP	00000336	RG	06	OPERMSGVEC	000026A4	R	04
CONNAME	000024D6	R	04	OPERNAMDESC	000026B8	R	04
CR	= 0000000D			OUTBUF	00002628	R	04
CRBSL_INTD	= 00000024			OUTBUF_STR	00002630	R	04
CTL\$GC_PCB	*****	X	05	OUTLEN	00002610	R	04
CTRSTR_AUTOLOG	000025F1	R	04	OUTLEN_UNIT	0000260C	R	04
CTRSTR_AUTOLOG_UNIT	00002600	R	04	PRS_IPC	= 00000012		
DDBSL_OCB	= 00000004			PUTERROR	*****	X	05
DDBST_DRVNAME	= 00000024			RIOSAB_BUFFER	*****	X	05
DYN\$C_LOADCODE	= 00000062			RIOSGW_OUTLEN	*****	X	05
EXESA_SYSPARAM	*****	X	04	RIOSOUTPUT_LINE	*****	X	05
EXESC_SYSPARS?	*****	X	04	SAVE_DOT	00002584	R	04
EXESGL_DEFFLAGS	*****	X	05	SBSB_SYSTEMID	= 00000018		
EXESGL_SCB	*****	X	06	SCH\$IOLOCKR	*****	X	05
EXESLINK_VEC	*****	X	06	SCH\$IOUNLOCK	*****	X	05
EXESLOAD_CODE	*****	X	06	SCS\$GA_LOCALSB	*****	X	05
EXESV_NOAUTOCNF	*****	X	05	SCS\$GL_CDL	*****	X	06
FACNAME	000024D0	R	04	SCS\$GL_MSCP	*****	X	06
FACNAMED	000024C8	RG	04	SELECT	000001D1	R	05
FACNAMSZ	= 00000006			SGN\$GET_DEVICE	000002C2	RG	05
FF	= 0000000C			SGN\$GET_DEVICE_LOCK_IDDB	000002D4	R	05
FULL_NAME_PTR	00002588	RG	04	SLV\$A_SYSVECS	= 00000010		
HELP_DESC	00002578	R	04	SLV\$B_TYPE	= 0000000A		
HELP_FILE	00002559	R	04	SLV\$L_INITRTN	= 00000004		
HELP_FLAG	00002574	R	04	SS\$DEVACTIVE	= 000002C4		
HLP\$M_PROMPT	= 00000001			SS\$DEVOFFLINE	= 00000084		
IDBSL_ADP	= 00000014			SS\$NOPRIV	= 00000024		
IOCSAUTOCONFIG	*****	X	05	SS\$NOSUCHDEV	= 00000908		
IOCSAUTORESET	*****	X	05	SYSCMEXEC	*****	GX	05
IOCSGL_ADPLIST	*****	X	05	SYSCMKRNL	*****	GX	05
IOCSSEARCHALL	*****	X	05	SYSSFAD	*****	X	05
IOGEN\$CONSOLE	*****	X	06	SYSG\$CONFQUAL	= 007C808A		
IOGEN\$LOADER	*****	X	05	SYSG\$INVADAP	= 007C80BA		
JPIS_PID	= 00000319			SYSG\$NOADAPTER	= 007C80D2		
LBR\$OUTPUT_HELP	*****	X	06	SYSG\$NOAUTOCNF	= 007C8002		
LF	= 0000000A			SYSG\$NODEV	= 007C9010		
LIB\$GET_INPUT	*****	X	06	TPA\$L_NUMBER	= 0000001C		
LIB\$PUT_OUTPUT	*****	X	06	TPA\$L_PARAM	= 00000020		
LINK_CODE	000005AE	R	06	TPA\$L_STRINGCNT	= 00000008		
LOADCODE	0000055D	R	06	TPA\$L_STRINGPTR	= 0000000C		
LOADRV	000004E3	R	06	TPA\$L_TOKENCNT	= 00000010		
LOCADP	000000DA	R	05	TPA\$L_TOKENPTR	= 00000014		
MMG\$A_SYSPARAM	*****	X	04	UBA_IDBASE	= 00001000		
MSCP_ARG_LIST	0000258C	R	04	UCB\$L_CRB	= 00000024		
MSCP_ARG_LIST_SIZE	= 00000030			UCB\$L_LINK	= 00000030		
MSCP_NAME	000025EC	R	04	UCB\$W_UNIT	= 00000054		
NEXTADP	000000B2	R	05	VALID_PAR_FILE	00002580	R	04
OPCSM_NM_CENTRL	= 00000001			VEC\$L_IDB	= 00000008		
OPCS_RQ_RQST	= 00000003						



+-----+  
! Psect synopsis !  
+-----+

PSECT name	Allocation	PSECT No.	Attributes
ABS	00000000 ( 0.)	00 ( 0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
\$ABSS	00000000 ( 0.)	01 ( 1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
\$\$\$000	00000000 ( 0.)	02 ( 2.)	NOPIC USR CON REL LCL NOSHR NOEXE RD NOWRT NOVEC BYTE
222	00000000 ( 0.)	03 ( 3.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC PAGE
NONPAGED_DATA	000027EB (10219.)	04 ( 4.)	NOPIC USR CON REL LCL NOSHR NOEXE RD WRT NOVEC QUAD
NONPAGED_CODE	0000039A ( 922.)	05 ( 5.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC LONG
PAGED_CODE	00000612 ( 1554.)	06 ( 6.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC LONG

+-----+  
! Performance indicators !  
+-----+

Phase	Page faults	CPU Time	Elapsed Time
Initialization	29	00:00:00.07	00:00:00.51
Command processing	110	00:00:00.77	00:00:02.09
Pass 1	571	00:00:23.37	00:00:47.02
Symbol table sort	0	00:00:03.60	00:00:07.09
Pass 2	276	00:00:05.53	00:00:09.68
Symbol table output	27	00:00:00.20	00:00:00.26
Psect synopsis output	2	00:00:00.04	00:00:00.07
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	1017	00:00:33.58	00:01:06.72

The working set limit was 1950 pages.  
132448 bytes (259 pages) of virtual memory were used to buffer the intermediate code.  
There were 130 pages of symbol table space allocated to hold 2263 non-local and 83 local symbols.  
1453 source lines were read in Pass 1, producing 119 object records in Pass 2.  
43 pages of virtual memory were used to define 40 macros.

+-----+  
! Macro library statistics !  
+-----+

Macro library name	Macros defined
\$255\$DUA28:[BOOTS.OBJ]BOOTS.MLB;1	1
\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	16
\$255\$DUA28:[SYSLIB]STARLET.MLB;2	20
TOTALS (all libraries)	37

2358 GETS were required to define 37 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS\$:CONFIGUTL/OBJ=OBJ\$:CONFIGUTL MSRC\$:CONFIGSW/UPDATE=(ENH\$:CONFIGSW)+MSRC\$:SYSGEN/UPDATE=(ENH\$:SYSGEN)+EXECML\$/LIB+LIB\$



0038 AH-BT13A-SE  
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION  
CONFIDENTIAL AND PROPRIETARY